ROBERTA SAMISTRARO TOMIGIAN

BLOCKCHAIN PARA GERENCIAMENTO DE CONFIANÇA EM REDES DTN

(*versão pré-defesa, compilada em 26 de janeiro de 2022*)

Trabalho apresentado como requisito parcial à conclusão do Curso de Bacharelado em Ciência da Computação, no Setor de Ciências Exatas, da Universidade Federal do Paraná.

Área de concentração: *Ciência da Computação*.

Orientador: Luis Carlos Pessoa Albini.

CURITIBA PR

2022

RESUMO

As redes tolerantes a atrasos e interrupções (*Delay and Disruption Tolerant Networks - DTNs*) são caracterizadas pela falta de conexão ponta-a-ponta. Para lidar com isso, elas adotam uma técnica de armazena-e-encaminha pela qual as mensagens são encaminhadas por meio de vários nós intermediários para chegar de uma origem a um destino. Durante o transporte dessas mensagens pode ocorrer alguns problemas de segurança, como a existência de nós maliciosos ou egoístas. Por isso, é importante que a rede possua um mecanismo de gerenciamento de confiança eficaz e seguro. Isso significa que ele deve realizar o cálculo de confiança e o armazenar em um banco de dados confiável. Esse trabalho propõe um modelo de gerenciamento de confiança distribuído para redes DTN e faz uso do *Blockchain* para realizar o armazenamento. Há a interação entre os nós por meio da troca de mensagens. No qual, para cada mensagem recebida, o nó destinatário recalcula o valor da confiança. O destinatário pode realizar esse cálculo acerca do remetente (direta). De outro modo, o destino recebe a tabela de confiança do remetente e calcula a confiança com base nos contatos dele (indireta). A proposta mantém o valor da confiança direta fixo e calcula apenas a oriunda de recomendação. Em seguida, é proposto um armazenamento no *Blockchain* da tabela do recebedor atualizada com os novos valores de confiança. O sistema mostrou a taxa de mensagens recomendadas com sucesso pelo sistema sofrem influência da troca de mensagens do remetente em relação ao destinatário. De modo que, praticamente todas as mensagens diretas são aceitas e há um filtro das mensagens de recomendação.

Palavras-chave: Redes tolerantes a atrasos. Gerenciamento de confiança. Blockchain.

ABSTRACT

Delay and Disruption Tolerant Networks (DTNs) are characterized by the lack of end-to-end connection. To deal with this, they adopt a store-and-forward technique whereby messages are forwarded through various intermediary nodes to arrive from a source to a destination. During the transport of these messages, some security problems may occur, such as the existence of malicious or selfish nodes. Therefore, it is important that the network has an effective and secure trust management mechanism. This means that it must perform the confidence calculation and store it in a trusted database. This work proposes a distributed trust management model for DTN networks and makes use of Blockchain to perform storage. There is interaction between nodes through the exchange of messages. In which, for each message received, the recipient node recalculates the trust value. The recipient can perform this calculation on the sender (direct). Otherwise, the destination receives the trust table from the sender and calculates trust based on the sender's contacts (indirect). The proposal keeps the direct trust value fixed and calculates only the one derived from the recommendation. Then, storage on the Blockchain of the recipient table updated with the new trust values is proposed. The system showed a rate of messages successfully recommended by the system affected by the exchange of messages from sender to receiver. All direct messages are accepted and there is a mode message filter.

Keywords: Delay tolerant networks. Trust management. Blockchain.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ACRÔNIMOS

DINF	Departamento de Informática
DTN	Delay/Disruption Tolerant Networks (Redes tolerantes a Atrasos/Interrupções)
E2E	End-to-end (Fim-a-fim)
HOP	Hop-by-hop (Salto-a-salto)
IDE	Integrated Development Environment (Ambiente de desenvolvimento integrado)
IoT	Internet of Things (Internet das Coisas)
IP	Internet Protocol (Protocolo da Internet)
MANET	Mobile Ad Hoc Network (Redes ad hoc móveis)
ONE	The Opportunistic Network Environment simulator (O simulador de ambiente de rede oportunista)
P2P	Peer-to-peer (Par-a-par)
TCP	Transmission Control Protocol (Protocolo de controle de transmissão)
UFPR	Universidade Federal do Paraná
VANET	Vehicular Ad Hoc Network (Rede ad hoc veicular)

SUMÁRIO

1 INTRODUÇÃO

De acordo com (Tanenbaum et al., 2011), redes de computadores são um conjunto de computadores autônomos interconectados por uma única tecnologia, e transmitem dados entre si. Essa conexão pode ser realizada via cabo ou sem fio. Para que essa comunicação seja realizável, isso é, ambos estejam na mesma rede e falem o mesmo idioma, é necessário ter um protocolo de rede.

Existem diversos protocolos, mas dois dos mais conhecidos são os de internet TCP/IP (*Transmission Control Protocol/Internet Protocol*). No qual o protocolo de transporte TCP é orientado a conexão, que é uma ligação constante entre dois dispositivos. Nele os dados trafegam de forma sequencial entre um dispositivo e outro, e geralmente a ordem de envio é preservada com a ordem de chegada dos dados. Isso garante a confiabilidade na entrega de dados fim-a-fim em cima de redes não confiáveis (Oliveira et al., 2007). Em contrapartida, de acordo com o (Tanenbaum et al., 2011), o protocolo IP visa realizar a interligação de redes, em que os roteadores IP encaminham cada pacote pela Internet, por um caminho de um roteador para o seguinte, até que o destino seja alcançado.

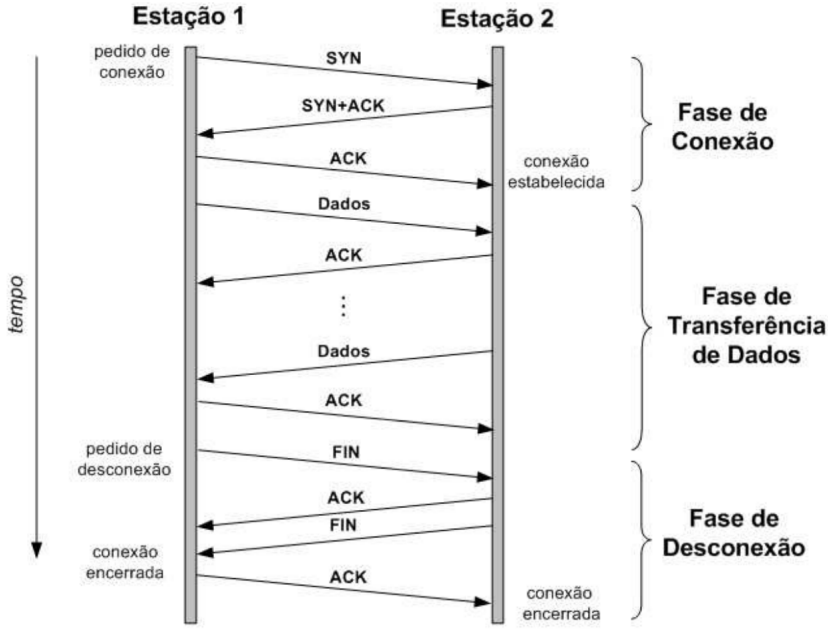


Figura 1.1: A fases de operação do TCP (Oliveira et al., 2007)

Na Figura 1.1, podemos perceber que a operação do TCP se dá pelas fases: estabelecimento de conexão, transferência de dados e desconexão. Em que, na fase de estabelecer a conexão é necessário fazer três trocas de mensagens (*three-way handshake - SYN, SYN-ACK e ACK*). Em seguida inicia-se a troca de dados, onde cada dado recebido pelo destino é sinalizado com reconhecimento positivo (*acknowledgments - ACKs*). Por fim, encerra-se a comunicação via

quatro mensagens (FIN, ACK, FIN e ACK). Podemos perceber que para esse protocolo funcionar corretamente é necessária a existência de um caminho fim-a-fim entre fonte e destino durante todo o período correspondente à sessão de comunicação, atrasos de comunicação relativamente pequenos e baixa taxa de erros (Oliveira et al., 2007).

Por isso, para os usos onde há frequentemente atrasos e desconexões (sem conexões fim-a-fim), as redes DTN (*Delay/Disruption Tolerant Networks*) são uma ótima solução. Elas são arquiteturas de rede desenvolvidas para lidar com conectividade intermitente e longos atrasos em redes sem fio. Elas utilizam a técnica comutação de mensagens, em que não há qualquer fase anterior ao envio de dados. Desse modo, quando é necessário enviar uma mensagem, ela é armazenada e enviada nó a nó desde a origem até o destino. Por esse fato, podemos dizer que as redes DTN são do tipo armazena-e-encaminha (*store-and-forward*). Isso significa que primeiramente a mensagem é integralmente recebida e armazenada para, em seguida, ser enviada ao próximo nó. Não é necessário, portanto, que o nó destino esteja ativo quando a origem envia a mensagem.

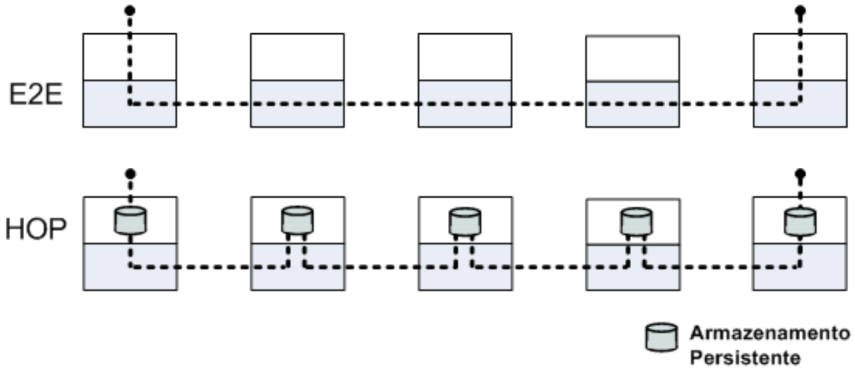


Figura 1.2: As configurações fim-a-fim (E2E) e salto-a-salto (HOP) (Oliveira et al., 2007)

Desse modo, na Figura 1.2, podemos ver duas configurações: fim-a-fim (E2E) e salto-a-salto (HOP). Na qual, a primeira poderia ser uma representação de comunicação da internet, onde há 5 nós dispostos em uma topologia linear em que três são intermediários, que realizam o encaminhamento, e os outros dois origem e destino. A segunda representação utiliza a comutação de mensagens onde cada nó armazena-e-encaminha a mensagem salto-a-salto.

A arquitetura é projetada para superar diversos desafios, permitindo a comunicação entre nós dentro de redes com características únicas nas quais o uso de protocolos de Internet é inviável.

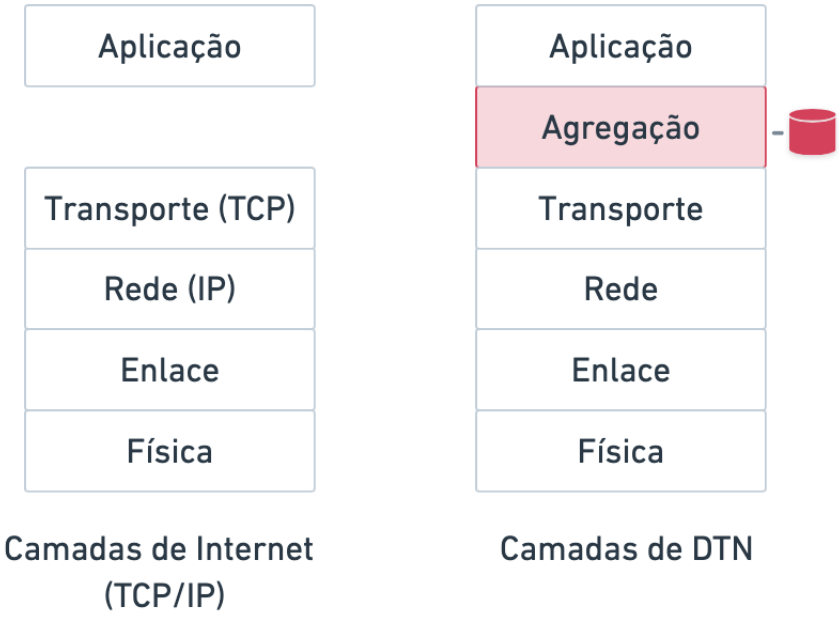


Figura 1.3: Comparação entre camadas TCP/IP e DTN.

Conforme podemos observar na Figura 1.3, a única diferença entre as camadas da rede DTN e TCP/IP é que na primeira há uma nova camada Agregação (*Bundle*). Por essa nova camada estar logo abaixo da Aplicação, isso significa que está acima das camadas específicas de cada região DTN, e com isso é comum a todas as regiões DTN. É possível, assim, conectar aplicações que estão em regiões diferentes e que possuem camadas inferiores com características distintas.

Outro ponto importante é o roteamento, que consiste na capacidade de transportar ou rotear dados de uma origem a um destino. Ele é um aspecto fundamental em qualquer tipo de rede de comunicação, incluindo a DTN. Os protocolos de roteamento na DTN podem ser classificados como cópia única e cópia múltipla. Em que no primeiro, as mensagens nunca são replicadas, ou seja, depois que a mensagem é entregue ao destinatário ela é deletada do buffer do remetente. Em contrapartida, o segundo caso, pode ser classificado em dois tipos: limitados e ilimitados. No qual, para o roteamento de múltiplas cópias limitado, a mensagem é replicada por um número de vezes e encaminhada aos outros nós da rede. Já para o ilimitado, não há um número fixo de replicações da mensagem na rede.

Nas redes DTN podemos considerar que os nós não são alcançáveis o tempo todo e podem entrar em comunicação com outros a qualquer instante. Por isso, temos o conceito de contato, que corresponde a uma ocasião em que é possível estabelecer uma conexão entre dois nós. Ela pode ser feita de diversos modos, e o tipo dela depende do tipo de operação da rede, que pode ser determinística, para redes interplanetárias, oportunísticas, para redes veiculares ou persistente, como na internet (Tornell et al., 2015). A explicação sobre alguns do tipos de contatos e exemplos e aplicações são apresentados na Tabela 1.1:

Tabela 1.1: Exemplos tipos de contatos. Adaptado de (Oliveira et al., 2007)

Tipo	**Descrição**	**Exemplo**
Persistente	Eles estão sempre disponíveis.	*Digital Subscriber Line* (DSL)
Sob demanda	Requer alguma ação para que ocorra, e uma vez que acontece a comunicação eles funcionam como o Persistente, até o encerramento.	Conexão discada
Programados	Uma agenda de contato é preestabelecida entre os nós antes da comunicação. O horário e a duração são definidos previamente entre eles.	Aplicações espaciais
Previsíveis	É possível que os nós façam previsões sobre o horário e a duração dos contatos com base em históricos anteriores. Mas há uma incerteza com relação a sua ocorrência, duração e horário.	Redes rurais
Oportunistas	Ocorre diante de encontro não programados entre os nós. Ele tem por objetivo se comunicar com nós que estão fora de alcance por intermédio de um terceiro.	*Pocket Switched Networks* (PSN)

1.1 APLICAÇÕES

O conceito de DTN foi inicialmente projetado para comunicação com espaçonaves, para compensar desconexões em distâncias Interplanetárias. No entanto, ao longo dos anos, os pesquisadores identificaram diversas outras áreas em que os conceitos de DTN podem ser empregados, como os apresentados na Tabela 1.2.

Tabela 1.2: Exemplos de aplicações DTN. Adaptado de (Rodrigues e Soares, 2015).

Aplicação	**Descrição**	**Exemplo**
Rede subaquáticas	Permitem aplicativos de coleta de dados oceanográficos, monitoramento de poluição, prevenção de desastres, navegação assistida e aplicativos de vigilância tática	(Cho et al., 2014)
Rede de rastramento de vida selvagem	Permitem monitorar o comportamento de longo prazo de animais selvagens distribuídos esparsamente em uma grande área	(Tovar et al., 2010)
Redes de recuperação de desastres	Podem oferecer suporte a comunicações entre equipes de emergência em áreas atingidas por catástrofes sem uma infraestrutura de comunicação funcional	(Kaisar, 2021)
Redes táticas militares	Podem facilitar as comunicações em ambientes hostis (campos de batalha) onde uma infraestrutura de rede não está disponível	(Amin et al., 2015)
Redes de pessoas	Permitem aplicativos que encaminham dados com base nos interesses sociais das pessoas	(Amendola et al., 2014)
Redes veiculares	Permitem o desenvolvimento de aplicações veiculares emergentes, como segurança no trânsito, monitoramento de tráfego, assistência ao dirigir e entretenimento informativo	(Agarwal et al., 2011)

1.2 CONTEXTUALIZAÇÃO

As redes tolerantes a atrasos e interrupções, pela falta de conexão ponta-a-ponta, adotam um esquema de armazenamento e encaminhamento pelo qual as mensagens são encaminhadas por meio de vários nós intermediários. Isso pode levar a diversos desafios, tais como: determinar qual será o próximo nó portador da mensagem, qual a confiabilidade de uma mensagem recebida e como lidar com a presença de nós egoístas ou maliciosos. (Li e Cao, 2011) definem que nós egoístas funcionam apenas para seu próprio ganho pessoal, eles encaminham apenas seus próprios pacotes e optam por não contribuir na rede. Nós maliciosos são os nós que tentam dificultar os parâmetros de rede, como taxa de entrega de mensagens e taxa de sobrecarga. Um exemplo da gravidade disso está em redes DTN veiculares, em que a ação desses nós ardilosos e egoístas podem causar perda de dados cruciais que podem causar ainda mais acidentes na estrada ou podem causar desperdício de recursos em DTN de restrição de recursos.

(Nagrath et al., 2019) apontam dois tipos de ataques feitos por nós maliciosos, de buraco negro e de inundação. No primeiro caso, o nó se torna atraente com o objetivo de pegar o

máximo de pacotes do nó em contato para em seguida descartar eles. O segundo caso pode ocorrer em redes DTN com restrição de recurso, em que os nós oblíquos tentam esgotar os recursos da rede inundando mensagens falsas ou inúteis na rede. Consequentemente, muitos nós retransmitem essas mensagens para outros membros da rede, o que desperdiça largura de banda, espaço de buffer limitado fornecido em nós DTN e aumenta o consumo de energia na rede. Diante disso, é necessário que a rede tenha um mecanismo de gerenciamento de confiança eficiente. A confiança é incorporada à decisão de roteamento para proteger a rede contra o mau comportamento malicioso do pacote, como, por exemplo, ataques de descarte (Asuquo et al., 2021).

1.3 OBJETIVOS

Esse trabalho propõe um mecanismo de gerenciamento de confiança para redes DTN. O gerenciamento é feito de forma distribuída com o uso da tecnologia *blockchain*, que visa armazenar o histórico de confiança. No qual, cada nó da rede possui um *blockchain* e nele guarda-se a relação que ele possui com outros nós e seus respectivos valores de confiança.

O uso da tecnologia *blockchain* para realizar um gerenciamento de confiança possui diversos benefícios em relação a segurança e confiabilidade. Um deles é que as transações são agrupadas em cada bloco por meio das funções *hash* criptográficas. Por esse motivo, é praticamente impossível alterar blocos armazenados anteriormente sem detecção. Isso já dificulta o ataque de nós maliciosos nos dados. Além disso, com o uso da criptografia de chave pública é possível verificar a autenticidade de blocos e transações.

Os principais objetivos desse trabalho são propor o modelo e a implementação de um gerenciamento de confiança indireta para redes DTN. Além disso, apresentar uma ideologia de como utilizar o *blockchain* para realizar o armazenamento, mas não será feita a implementação e validação. Finalmente, espera-se corroborar a legitimidade do cálculo da confiança, analisar a influência da confiança direta na indireta e os fatores que interferem na eficiência do modelo proposto.

1.4 ESTRUTURA DA MONOGRAFIA

Esta monografia está estruturada em 7 capítulos. O Capítulo 2 apresenta o conceito de gerenciamento de confiança. Capítulo 3 apresentam os conceitos necessários para a compreensão do desenvolvimento da pesquisa. O Capítulo 4 discorre sobre os trabalhos relacionados. O Capítulo 5 descreve como foi projetado e implementado o mecanismo proposto. O Capítulo 6 apresenta a avaliação da implementação proposta com relação ao mecanismo. Finalmente, o Capítulo 7 apresenta as considerações finais sobre o estudo realizado e sugestões de trabalhos futuros.

2 GERENCIAMENTO DE CONFIANÇA

O conceito de confiança está diretamente associado a verdade, pode ser definido como "acreditar na sinceridade de alguém"ou "segurança em relação a alguma coisa". Em outras palavras, ela é expressa em termos de um relacionamento entre alguém que deposita sua confiança em uma entidade alvo (Grandison e Sloman, 2000).

A confiança é um fator importante para a tomada de decisão e para a manutenção de um relacionamento baseado na colaboração e cooperação. A quantificação da confiança na computação tornou-se mais complicada uma vez que é necessário derivá-la de redes complexas e composta de várias camadas, tais como protocolos de comunicação, trocas de informações, interações sociais e motivações cognitivas. Existem diversos aspectos que afetam a definição dela, que podem ser representados pela Figura 2.1 (Cho et al., 2015).

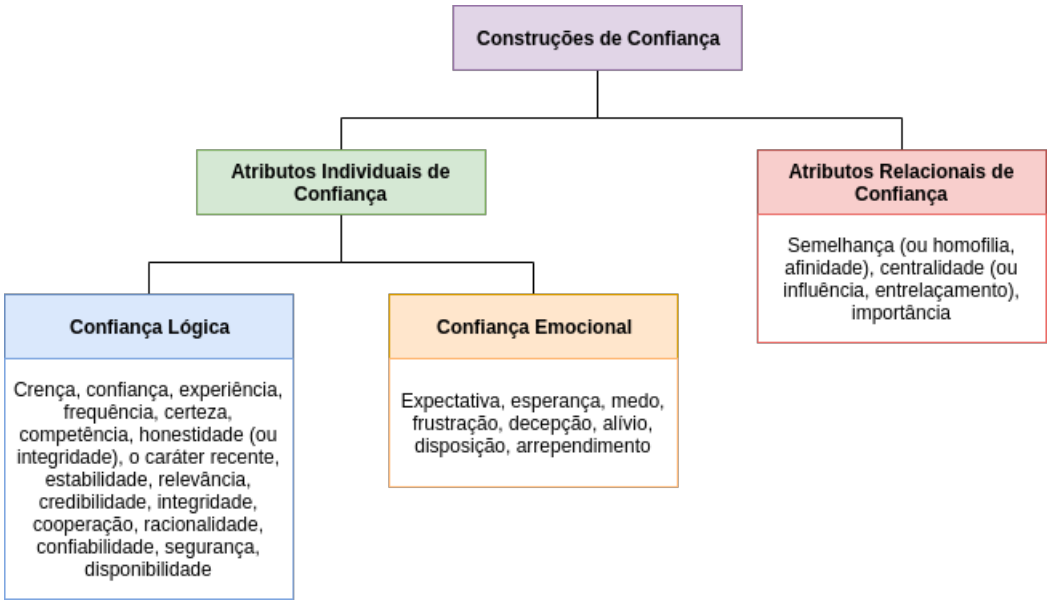


Figura 2.1: Classificação de fatores que afetam a confiança (Cho et al., 2015)

Há dois tipos de atributos para a construção da confiança, Individuais e Relacionais. O primeiro pode ser derivado de uma característica lógica, como credibilidade ou disponibilidade, ou de uma característica emocional, como uma frustração. Já o segundo, pode ser definido a partir de uma importância, afinidade ou influência.

Adicionalmente, os Atributos Individuais de confiança podem ser propriamente relacionados a valores obtidos de um indivíduo relativo ao outro. Isso é, um indivíduo define sua confiança em relação ao outro de acordo com suas experiências interpessoais (Seção 2.1).

Em contrapartida, os Atributos Relacionais de confiança podem ter a ver com valores obtidos por defluência de outro acerca de um terceiro indivíduo. Um indivíduo define sua confiança em relação a outro de acordo com experiências de terceiros (Seção 2.2).

Dentro desse contexto, há o Gerenciamento de Confiança que, de acordo com (Blaze et al., 1996), é: *"Uma abordagem unificada para especificar e interpretar políticas de segurança, credenciais e relacionamentos que permitem a autorização direta de ações críticas de segurança"*.

O Gerenciamento de Confiança, nesse sentido, lida com um projeto de políticas eficazes e obrigatórias. Além de métodos para conceder permissões a usuários de confiança, os componentes que podem assegurar essas políticas e permissões para proteger e administrar os recursos de sistema. Esse conceito, aplicado à Redes de Computadores pode ser construído de forma Centralizada, Descentralizada ou Distribuída (Seção 2.3).

2.1 CONFIANÇA DIRETA

De acordo com (Mu e Yuan, 2010), a confiança possui três dimensões, na qual a primeira é o originador da confiança, a segunda é o propósito de confiança e a terceira é o alvo da confiança. A confiança está diretamente relacionada com o propósito da confiança. Considere, por exemplo, o cenário em que temos um desenvolvedor X que trabalha em uma empresa Y, em que Y confia em X para trabalhar em seu setor, de desenvolvimento web. Mas Y não confia em X para exercer seu papel no setor de finanças. Deste modo, podemos perceber que o originador da confiança seria Y, o alvo X e o propósito seria em qual setor o alvo tem maior chance de trabalhar.

Conforme foi apresentado no Capítulo 2, a definição de confiança depende de alguns atributos. Nessa Seção serão tratados os Atributos Individuais de Confiança Lógica e Emocional.

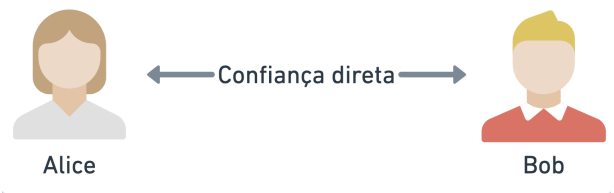


Figura 2.2: Representação de confiança direta

Podemos ver um exemplo na Figura 2.2, onde temos que Alice e Bob compartilham de uma confiança direta. Na qual, a confiança de um no outro foi oriunda de um propósito, como por exemplo na Tabela 2.1.

Tabela 2.1: Exemplos de confiança direta

Nome	**Tipo de confiança**	**Descrição**
Confiabilidade	Lógica	Indica confiança em uma fonte de informação em que a fonte exibe comportamento consistente.
Experiência	Lógica	É medida a experiência com base no fato de a expectativa da entidade ser atendida ou não, o que determina a experiência positiva ou negativa.
Medo	Emocional	É um sentimento de esperar um resultado negativo em uma situação incerta, onde um objetivo importante deve ser alcançado.
Frustração	Emocional	É um sentimento que ocorre quando uma expectativa positiva acabou por falhar.

O valor de confiança de Alice em Bob e de Bob em Alice pode ter sido determinado por pelo menos um dos fatores de confiança listados acima, como a confiabilidade ou medo, por exemplo. Para cada um deles há um tipo específico de cálculo, que são especificados em (Cho et al., 2015), assim como diversos outros fatores que podem ser levados em consideração.

2.2 CONFIANÇA INDIRETA

Conforme foi apresentado no Capítulo 2, a definição de confiança depende de alguns atributos. Nessa Seção serão tratados os Atributos Relacionais de Confiança, que está diretamente relacionado à Confiança Indireta. Ela ocorre entre dois indivíduos mediante um terceiro.

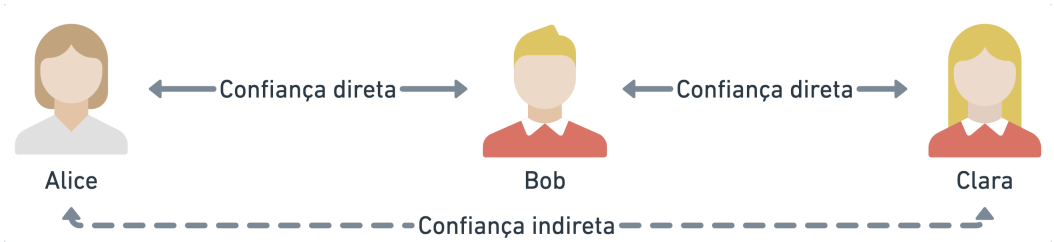


Figura 2.3: Representação de confiança indireta

Podemos observar um exemplo na Figura 2.3, em que temos que Alice e Clara confiam diretamente em Bob, mas não possuem relação direta entre si. Nesse caso, elas obtiveram uma Confiança Indireta por intermédio de Bob.

2.3 ARQUITETURAS DE CONFIANÇA

O Gerenciamento de Confiança aplicado em Redes de Computadores seguem os 3 tipos de estruturas de redes: Centralizado, Descentralizado e Distribuído. Cada um deles possui

características que definem qual seria a melhor abordagem dependendo do tipo de aplicação. Na Figura 2.4 temos uma representação desses tipos diferentes de arquiteturas.

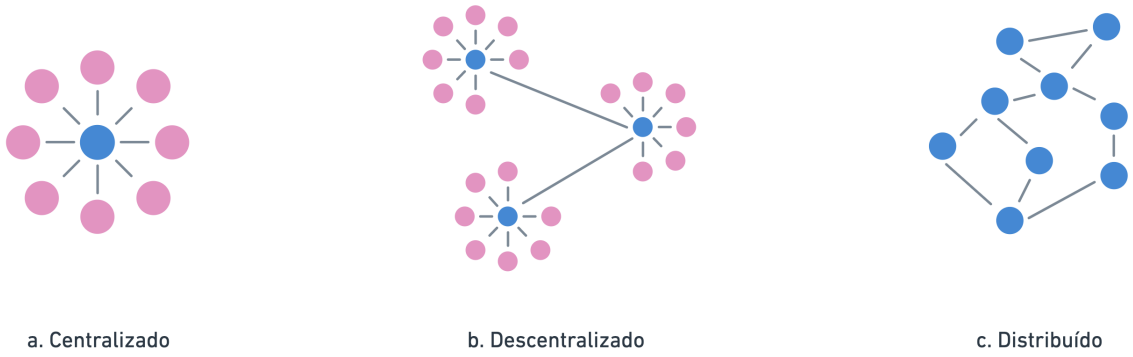


Figura 2.4: Comparação dos tipos de redes

Na Figura 2.4 (a), temos a confiança centralizada. Na qual cada solicitação ou serviço de confiança passará por um nó central. Ele pode ser acessado por todos os outros nós de seu domínio e é responsável por: gerenciar os dados de confiança, calcular, tomar decisão e auxiliar os outros nós fornecendo as informações necessárias de confiança (Truong et al., 2016). Na confiança descentralizada, Figura 2.4 (b), não há um nó central único. Temos vários blocos de nós centralizados que conectam-se entre si pelos nós centrais. Por fim, a confiança distribuída Figura 2.4 (c), em que não há uma entidade centralizadora. Nesse método cada nó está conectado diretamente com outro nó ou rede participante, que ele próprio é responsável pelo gerenciamento, utilizando cálculos de Confiança Direta (Seção 2.1) ou Confiança Indireta (Seção 2.2).

Tabela 2.2: Comparações entre arquiteturas (Truong et al., 2016)

Propriedade/ Comportamento	**Centralizado**	**Descentralizado**	**Distribuído**
Pontos de falha	Um ponto	Finitos	Infinitos
Manutenção	Fácil	Moderado	Difícil
Estabilidade	Baixa	Média	Alta
Escalabilidade	Baixa	Alta escalabilidade, capacidade finita	Infinito
Complexidade de criação	Baixa	Moderada	Mais detalhes necessários
Evolução/ Diversidade	Baixa	Alta	Alta

Além disso, podemos observar e comparar algumas propriedades de cada um dos métodos na Tabela 2.2. Podemos perceber que o método centralizado, por possuir uma única entidade centralizadora, possui um risco alto em relação a tolerância a falhas, pois se esse nó

falhar, compromete a rede inteira. Em contrapartida, o método descentralizado, por possuir mais de ponto central, diminui um pouco essa vulnerabilidade, pois se um nó falhar, a rede continua. Esse problema é menor ainda nas redes distribuídas e possui um risco baixíssimo de vulnerabilidade por falha.

Outra questão interessante é com relação à Evolução e Escalabilidade. O centralizado evolui mais lentamente e também possuem baixa escalabilidade. Já o descentralizado, por possuir mais pontos de acesso, consegue evoluir mais rápido que o primeiro e tem uma alta escalabilidade com capacidade limitada. Por fim o distribuído, que possui ainda mais pontos de acesso, o que torna a evolução muito mais rápida que o descentralizado e ainda tem escalabilidade infinita.

Com relação a complexidade de criação e atualização da rede, o centralizado é o mais fácil e rápido de lidar. Em seguida está o descentralizado, que é um pouco mais lento. Aditivamente o distribuído, que é o mais lento de todos. Por fim, a manutenibilidade, que é muito fácil no centralizado pelo fato de ter somente um ponto de falha. Seguido por descentralizado, que possui mais pontos de falhas que o centralizado, mas menos que os distribuídos, que são os mais difíceis de manter devido ao grande número de nós.

2.4 RESUMO

Este capítulo apresentou os fundamentos de Gerenciamento de Confiança. O conceito de confiança está ligado a sentir segurança em relação a alguma coisa ou alguém. Há dois tipos de atributos para a construção da confiança os Individuais e os Relacionais, que estão respectivamente ligados a Confiança Direta e Indireta. Na Confiança Direta, como o próprio nome diz, um indivíduo define o valor da confiança em relação ao outro com base nas experiências interpessoais. Diferentemente da Confiança Indireta, que é definida com base em experiência de terceiros. O Gerenciamento de Confiança aplicado em redes de computadores pode ter uma estrutura Centralizada, Descentralizada ou Distribuída. A Centralizada caracteriza-se por uma unidade central que é responsável por todo o gerenciamento da rede. A Descentralizada contém vários blocos de nós centralizados que se conectam. Por fim, a Distribuída, na qual cada nó é responsável pelo gerenciamento de confiança.

3 BLOCKCHAIN

O *blockchain* é um *ledger* (livro-razão) distribuído e imutável. Ele se tornou muito conhecido por ser a tecnologia que viabilizou a rede da criptomoeda *Bitcoin*. Esse conceito foi desenvolvido pelo indivíduo de pseudônimo Satoshi Nakamoto. Para (Nakamoto, 2008), o *blockchain* é uma rede que registra as transações em uma cadeia contínua baseada em *hash*, que forma um registro que não pode ser alterado sem refazer o trabalho.

Uma função *hash*, representada pela Figura 3.1, é uma função que tem uma entrada de caracteres de tamanho variável e uma saída de tamanho fixo. No contexto da criptografia, uma boa função *hash* deve possuir, dadas duas entradas diferentes, uma baixa probabilidade de resultar em duas saídas iguais. Outra propriedade importante é a dificuldade de se reverter a função, ou seja, encontrar a entrada x a partir da saída h(x).

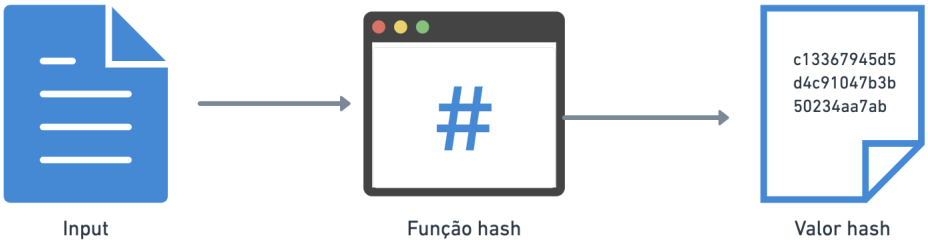


Figura 3.1: Função *hash*

O *blockchain* é uma rede *peer-to-peer* (P2P), em que os computadores podem interagir uns com os outros sem passar por um computador servidor central, como mostra a Figura 3.2. Por não possuir a necessidade de um servidor central, a rede possui uma alta flexibilidade, permitindo a entrada e saída de nós sem que suas operações sejam interrompidas. Assim, no P2P, os nós são participantes são igualmente privilegiados e equipotentes na aplicação que a rede executa, e cada nó possui o papel tanto de cliente quanto de servidor (Bauwens et al., 2019). Uma característica muito importante das redes P2P é a sua capacidade de tornar os seus registros inalteráveis e praticamente inextinguíveis, pois os dados de todas as transferências feitas se espalham por todos os nós da rede (Kouicem et al., 2020).

Para entender o funcionamento do *blockchain*, suponha que uma pessoa deseja realizar uma transação como, por exemplo, transferir um valor x para outra pessoa. Seguindo como base a Figura 3.3, podemos perceber que uma transação (Seção 3.2) é criada e submetida à rede *blockchain*, que a princípio não reconhece ela como válida. Em seguida, ocorre o processo de validação (Seção 3.3) da transação pelos mineradores, que realizam uma computação para rodar o algoritmo de consenso (Seção 3.4). Em seguida, várias transações válidas são agrupadas em um novo bloco (Seção 3.1), que é adicionado à rede *blockchain*, que reconhece esse novo bloco. Depois disso, a transferência é confirmada.

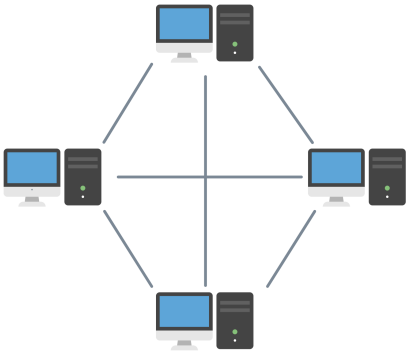


Figura 3.2: Arquitetura P2P

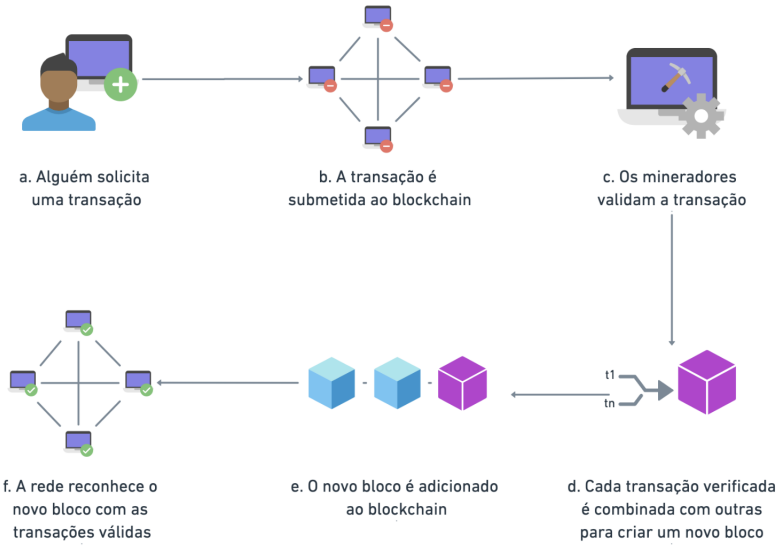


Figura 3.3: Funcionamento do *blockchain*

3.1 BLOCO

O *blockchain* consiste em uma base de dados, que é composta por uma cadeia de pacotes de dados, onde um bloco compreende múltiplas transações (tx). Além disso, ele possui um carimbo de data e hora (*timestamp*), a *hash* do bloco anterior e um Nonce, que em criptografia é um número aleatório que só pode ser utilizado uma vez e é utilizado para verificar a *hash*. Esse conceito garante a integridade da cadeia até o primeiro bloco (Nofer et al., 2017). Uma representação dessa estrutura pode ser vista na Figura 3.4.

Como podemos observar, cada transação do *blockchain* possui um *hash* que permite identificá-la. A partir dessa propriedade, é possível a implementação das árvores de Merkle, de acordo com (Okupski, 2014), é uma estrutura de dados que armazena um resumo dos dados de cada transação. Em que transação é representada por uma folha, e cada folha possui, portanto, um *hash* único. O *hash* de um nó pai é gerado a partir da concatenação do *hash* de seus dois nós filhos, submetida a uma nova rodada de *hash*. Esse procedimento acontece sucessivamente até a raiz da árvore ser gerada.

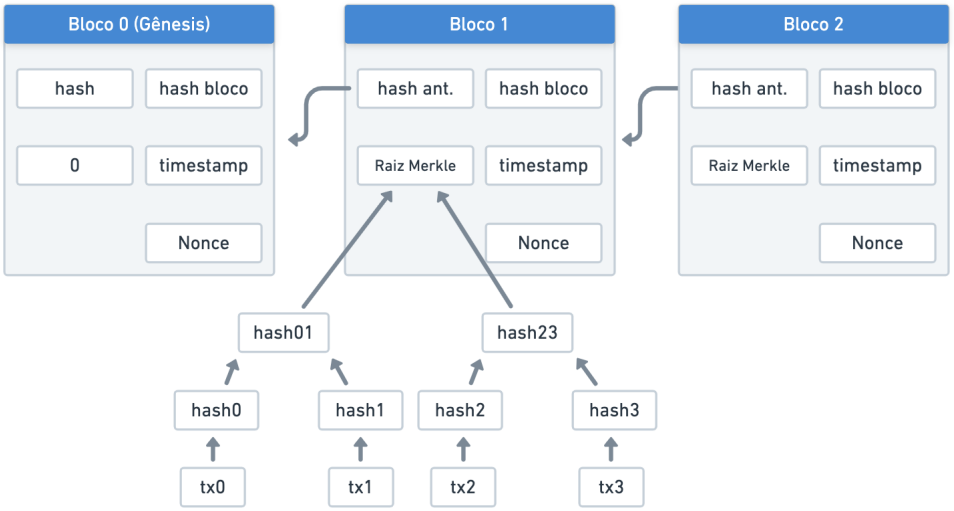


Figura 3.4: Estrutura do bloco. Adaptado (Xu et al., 2020)

3.2 TRANSAÇÃO

Uma transação no *blockchain* pode ser definida como uma pequena unidade de tarefa que é armazenada em um bloco. A Figura 3.5 mostra um exemplo de como é a estrutura de uma transação de um remetente para um destinatário na moeda *Bitcoin*. Podemos perceber ela é dividida em dois campos, a lista de entradas e a lista de saídas. Cada uma das entradas especificam as moedas não gastas de um determinado usuário e possuem: o identificador da transação anterior e uma assinatura digital do usuário, que é um método de autenticação que imita as funções de uma assinatura real a partir da criptografia. Cada uma das saídas representam a quantia transferida para o endereço especificado de um destinatário e possuem: o valor a ser transferido para o destinatário e a chave pública dele, que é utilizada para verificar a permissão de acesso dessas saídas por parte de entradas de outras transações. A saída de uma transação é usada como entrada da próxima transação para verificação. A verificação da transação é distribuída para os nós da rede P2P e apenas as transações válidas são registradas. Para autorizar o gasto das moedas na entrada da transação, o usuário deverá apresentar sua assinatura para a transação e a chave pública correspondente (Park et al., 2018).

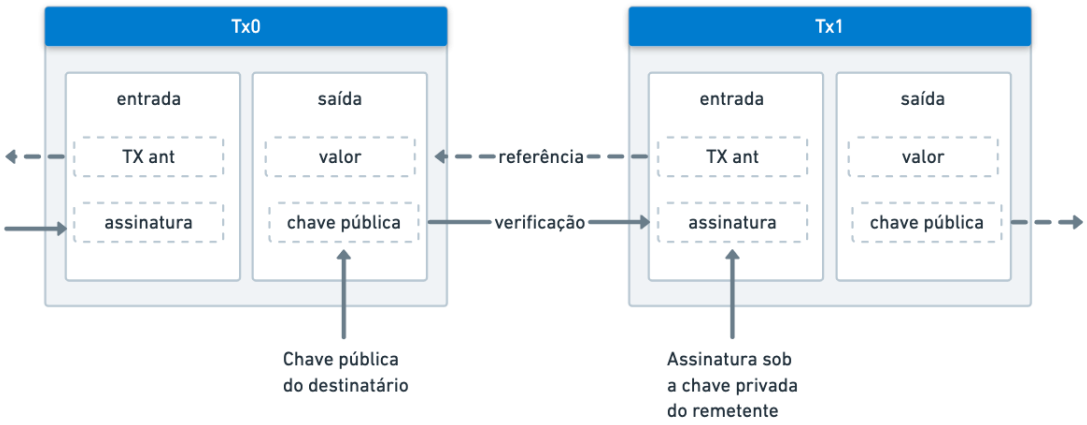


Figura 3.5: Estrutura da transação do *Bitcoin* (Park et al., 2018)

3.3 MINERAÇÃO

Todas as transações feitas no *blockchain* devem passar por um processo de validação antes de serem efetivamente inseridas na cadeia. Esse processo, chamado de mineração, tem o propósito de garantir a segurança do *blockchain* e recompensar os mineradores, como são chamados os nós da rede que executam esse procedimento. O minerador recebe esse incentivo por meio de moedas para cada bloco validado e o somatório das taxas incluídas em todas as transação do bloco. Por haver uma recompensa para o minerador por cada bloco validado, esse sistema estimula a competitividade (Bhaskar e Chuen, 2015).

Quando uma transação é feita, ela é coletada pelo minerador. Para o minerador conseguir validar o bloco recebido, é necessário que seja feito um esforço computacional, chamado de *Proof-of-Work (POW)*, para garantir o seu mérito. Esse esforço é necessário, pois é uma maneira de decidir qual é o nó que tem o direito de validação de determinado bloco.

Caso haja uma mineração de um mesmo bloco concluída por dois nós em uma curta diferença de tempo, a questão é resolvida pela Corrida de Blocos. A Corrida de Blocos define que o minerador que insere o bloco no maior segmento da cadeia recebe o direito de validação. Todos os nós inserem os blocos em um segmento, mas quando há bifurcação, o menor segmento é invalidado, pois possuem os chamados de blocos órfãos.

3.4 ALGORITMO DE CONSENSO

Em 1982 surgiu o desafio, proposto por Marshall Pease, Robert Shostak e Leslie Lamport, dos generais bizantinos. Esse problema consiste em: dados dois generais e seus respectivos exércitos, cercando uma cidade, devem determinar em que momento ocorrerá o ataque com o detalhe de que eles não tem como se comunicar. Exceto via mensageiro, que corre o risco de ser encontrado por tropas rivais. Entretanto, esse método, caso realizado com sucesso, não transmite segurança para os generais, pois não há a validação de que o aliado recebeu o recado e se irá, por exemplo, atacar no horário combinado. Como garantir o consenso entre eles, distribuídos nos

arredores da cidade é semelhante com as objeções que têm-se para validar operações em sistemas distribuídos, que "é um conjunto de computadores independentes entre si que se apresenta a seus usuários como um sistema único e coerente" (Tanenbaum et al., 2011).

Com o propósito de encontrar maneiras de superar esse obstáculo, soluções foram inventadas e para elas deu-se o nome de Algoritmos de Consenso. São alguns desses métodos: *Proof-of-Work (POW)*, *Proof-of-Stake (PoS)* e o *Proof of Useful Work (uPoW)*. Considere, por exemplo, o algoritmo POW. Ele é utilizado no *Bitcoin* e *Ethereum*, onde é necessário que o minerador encontre um valor chamado *nonce*, cujo *hash* com a raiz de árvore de Merkle do bloco seja menor ou igual que um parâmetro x. Quando um minerador consegue resolver esse desafio, o bloco é validado e inserido na cadeia, e o minerador recebe uma quantia em *Bitcoins*. O POW é um desafio com base na criptografia, que tem o intuito de fazer com que com os nós de uma rede executem uma certa quantidade de trabalho para garantir o seu direito de realizar determinado processo. Normalmente esse desafio consiste em resolver problemas matemáticos com um alto nível de dificuldade e que exijam um alto poder computacional para serem resolvidos. Como o processamento computacional resulta em um alto gasto energético, esse procedimento acaba reduzindo as chances de qualquer ataque cibernético, pois acaba sendo desvantajoso (Pires, 2016).

3.5 APLICAÇÕES

A Figura 3.6 mostra como foi o desenvolvimento das gerações do *blockchain* ao longo dos anos. A tecnologia *blockchain* foi proposta pela primeira vez para oferecer suporte a criptomoedas como *Bitcoin*, portanto, *blockchains* de criptomoedas e aplicativos relacionados são frequentemente rotulados como *Blockchain 1.0*. A criação de contratos inteligentes e sua união com as moedas digitais, criou novas aplicações financeiras que são classificadas como *Blockchain 2.0*. Entretanto, a cadeia de blocos não se limita apenas ao mercado financeiro, que é apenas uma possível implementação do conceito mais amplo de *Distributed Ledger Technology (DLT)*. Na verdade, o livro-razão distribuído pode conter qualquer tipo de informação, não precisa ser necessariamente relacionada a finanças. As aplicações que se referem ao espectro mais amplo de usos dos livros-razão distribuídos não relacionados à criptomoedas, são comumente chamadas de aplicações *Blockchain 3.0* (Maesa e Mori, 2020). Por fim, o *Blockchain 4.0* se concentrou principalmente em serviços como livro-razão público e bancos de dados distribuídos em tempo real. Este nível tem integração perfeita de aplicativos baseados na Indústria e Saúde 4.0 (Bodkhe et al., 2020). Alguns exemplos de aplicações do *blockchain* estão representado na Tabela 3.1.

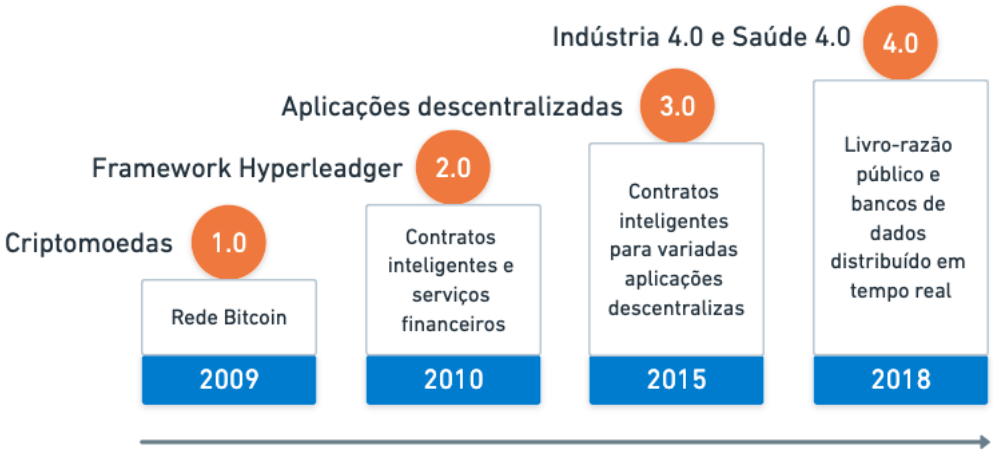


Figura 3.6: Gerações do *blockchain*. Adaptado de (Bodkhe et al., 2020)

Tabela 3.1: Exemplos de aplicações do *blockchain*. Adaptado de (Nofer et al., 2017) e (Maesa e Mori, 2020)

Aplicação	**Descrição**	**Exemplo**
Criptomoedas	Redes e meios de troca usando criptografia para proteger as transações	(Nakamoto, 2008)
Seguro	Propriedades podem ser registradas no blockchain e as seguradoras podem verificar o histórico de transações	(Zhou et al., 2018)
Industria musical	Determinar os royalties da música e gerenciar a propriedade dos direitos musicais	(O'Dair e Beaven, 2017)
Existência de documento	Armazenar e validar a assinatura e o timestamp de um documento usando blockchain	(Fill e Härer, 2020)
Armazenamento descentralizado	Compartilhamento de documentos sem a necessidade de terceiros	(Jiang et al., 2020)
Internet das coisas	O blockchain armazena de forma confiável a comunicação de dispositivos inteligentes na internet das coisas	(Yazdinejad et al., 2020)
Soluções antifalsificação	A autenticidade dos produtos é verificada pela rede blockchain	(Lee e Yeon, 2021)
Aplicativos de internet	Em vez de governos e corporações, os servidores de nomes de domínio (DNS) são controlados por todos os usuários de forma descentralizada	(Karaarslan e Adiguzel, 2018)
Voto eletrônico	Qualquer tipo de sistema de votação em que os votos são emitidos e computados por meio de sistemas eletrônicos.	(McCorry et al., 2017)
Saúde	O blockchain gerencia e armazena dados sensíveis de saúde de forma segura	(Ekblaw et al., 2016)
Sistema de gerenciamento de identidade	Um sistema de identidade centrado no usuário é um sistema federado em que o usuário tem controle de seus dados de identidade	(Naik e Jenkins, 2020)
Sistema de controle de acesso	O blockchain pode ser usado para armazenar políticas de controle de acesso, solicitações de acesso e os resultados de permissões.	(Maesa et al., 2017)

3.6 RESUMO

Este capítulo apresentou os fundamentos do *Blockchain*, que é um livro-razão de arquitetura distribuída. Ele é constituído por uma cadeia de blocos, no qual cada um contém um conjunto de transações. Todas as transação no *Blockchain* passam por um processo de validação antes de serem inseridas na cadeia. Esse processo é chamado de mineração e tem como objetivo garantir a segurança do *Blockchain* e recompensar os nós da rede que executam o procedimento. Essa validação ocorre mediante a um esforço computacional, que se chama Algoritmo de Consenso. Ele consiste em resolver problemas matemáticos com um alto nível de dificuldade e gasto de energia.

4 TRABALHOS RELACIONADOS

Este capítulo aborda os trabalhos relacionados considerados para a pesquisa e o desenvolvimento da solução proposta. A metodologia de pesquisa consistiu em reunir estudos no contexto de gerenciamento de confiança, redes DTN e *blockchain*.

O trabalho (Wu e Liang, 2021) apresentou um gerenciamento de confiança baseado em *blockchain* para a Internet das Coisas (IoT) com recursos limitados, onde a confiabilidade dos nós sensores será avaliada pelos nós de borda móveis. O cálculo de confiança é publicado por um nó sensor como a tarefa. Os nós de borda móveis podem competir pela tarefa. A tarefa pode ser concluída várias vezes por nós de extremidades móveis diferentes. As operações de publicação, recebimento e acabamento são vistas como as transações que são adicionadas aos blocos e verificadas pelos mineiros. Os resultados de computação de confiança de nós de extremidade móveis são avaliados por contratos inteligentes no *blockchain*, em vez de um terceiro subjetivo. Portanto, nós de borda móveis mal-intencionados não afetam o gerenciamento de confiança. Além disso, resolve o problema dos recursos escassos pois um nó de borda móvel é o nó com capacidade de computação e armazenamento relativamente forte.

O trabalho de (Roy et al., 2021) visa analisar a possibilidade de integrar a tecnologia *blockchain* com redes DTN aplicadas no cenário de uma gestão de desastres. O mecanismo desenvolve uma forma alternativa de transmitir e autorizar transações de *blockchain* sem ter que depender de conectividade com a Internet. No cenário proposto pelos autores foram idealizadas zonas compostas por nós *Shelter*, responsáveis por gerar e enviar as necessidades do abrigo aos *Volunteers*. Eles se movimentam pela área atingida, coletam as necessidades do abrigo, trocam as necessidades entre si via protocolo de roteamento de rede DTN e transmitem os dados aos *Data Mules*. Os *Data Mules* são veículos de abastecimento que coletam as necessidades de abrigo dos voluntários, processam, criam os blocos e fazem o *upload* no *blockchain* local. Após isso, os nós de retransmissão estacionários, que possuem conexão com a internet, acessam o *blockchain* local e publicam o bloco criado no *blockchain* global. Isso possibilita que os *Stakeholders* tenham acesso às necessidades daquela região. Foi provado pelos autores que é possível utilizar *blockchain* em uma rede tolerante à falta de conexão com o uso do recurso armazena-e-encaminha entre os nós em conjunto com nós de retransmissão estacionários.

O artigo (Greca e Albini, 2018) propõe um modelo para gerenciamento de confiança em redes veiculares. Com esse modelo, os nós da rede podem determinar quais nós são dignos de confiança ou não. Em que, cada nó de uma rede armazena um grafo direcionado de confiança, que utiliza o conceito de componentes fortemente conexos e de coloração de grafos para detectar nós maliciosos em uma rede. Cada nó u da rede armazena um grafo direcionado de confiança $T = (V, E)$. No qual cada nó em V representa um membro da rede e cada aresta E representa a confiança entre os dois nós. De acordo com os autores, o modelo é muito eficiente, é um dos

poucos que apresenta a complexidade e é capaz de detectar nós que começam benignos e se tornam maliciosos durante a simulação.

(De Andrade e Albini, 2016) sugerem um esquema de gerenciamento de chaves para redes DTN baseado em cadeias de assinaturas digitais. Em que os nós em intervalos de tempo enviam aos seus vizinhos uma mensagem contendo a lista de cadeias. Suponha um nó z que recebe a cadeia no formato $((C(x))...)_y$, no qual z pode descobrir o último nó que assinou (y) e a origem da cadeia (x). Caso z tenha confiança em y, ele assina por cima e repassa a cadeia para seus vizinhos no mesmo padrão $((C(x))...)_z$. Uma cadeia é considerada aberta quando o nó conseguiu verificar todas as assinaturas e obter o certificado de origem e fechada caso o nó não tenha conseguido validar uma assinatura. Segundo os autores, pelo fato desse modelo ser descentralizado, consegue garantir uma segurança por ter vários pontos de falha, funciona bem em comunicações descontínuas e é seguro contra vários tipos de ataques.

4.1 DISCUSSÃO

Os trabalhos que serviram de base para esta pesquisa estão na Tabela 4.1, ordenados por similaridade com a solução proposta e correlacionadas as suas abrangências em relação aos assuntos gerenciamento de confiança (GC), redes DTN (RD) e *blockchain* (B). Nenhum deles inclui gerenciamento de confiança em redes DTN utilizando a tecnologia *blockchain*, o que demonstra a principal contribuição desta pesquisa.

Tabela 4.1: Trabalhos sobre gerenciamento de confiança, redes DTN e blockchain.

Artigos	**Objetivo**	**Abordagens**		
		GC	**RD**	**B**
A blockchain-based trust management method for Internet of Things (Wu e Liang, 2021)	Propor um gerenciamento de confiança baseado no blockchain para a IoT	Direta e indireta	-	Global
Blockchain Leveraged Node Incentivization in Cooperation-Based Delay Tolerant Networks (Roy et al., 2021)	Propor a utilização do blockchain para o gerenciamento de desastres em redes DTN	-	Todas	Global
TruMan: Trust Management for Vehicular Networks (Greca e Albini, 2018)	Propor um gerenciamento de confiança para redes veiculares	Direta e indireta	Veiculares	-
Fully distributed public key management through digital signature chains for delay and disrupt tolerant networks (De Andrade e Albini, 2016)	Fornecer um gerenciamento de assinatura de chaves para redes DTN	-	Todas	-

4.2 RESUMO

Este capítulo discutiu os trabalhos existentes na literatura relacionados aos seus respectivos temas. Além de realizar uma comparação das abordagens de cada um deles com relação ao sistema proposto nesse trabalho.

5 BLOCKCHAIN PARA GERENCIAMENTO DE CONFIANÇA EM REDES DTN

A proposta desse trabalho consiste em um sistema que utilizar o *Blockchain* para gerenciamento de confiança em uma rede DTN. O cálculo da confiança de um nó em relação ao outro está definido na Seção 5.1. Por fim, a Seção 5.2 apresenta detalhes de implementação do sistema.

5.1 CÁLCULO DA CONFIANÇA

O cálculo da confiança entre dois indivíduos define um valor para quantificar essa relação social. Esse trabalho implementa um Gerenciamento de Confiança Indireta (Seção 2.2) que foi inspirado no modelo proposto por (Misaghi et al., 2012), em que os nós formam cadeias de confiança com base em evidências de comportamento mantidas na rede de confiança e usam essas cadeias para estimar a confiabilidade de outros nós. Os nós trocam periodicamente as informações armazenadas em suas redes de confiança com outros nós de confiança, fornecendo um método para realizar recomendações com baixo custo de comunicação. Diferentemente do (Misaghi et al., 2012), nesse trabalho é considerado apenas um nível para o cálculo da confiança. Suponha que n_i possua um certo valor de confiança em n_j, mas não possua confiança em n_u. Para estimar a confiança de n_i em n_u, deve-se realizar o produto entre a confiança de n_i em n_j (T_{ij}) pela confiança entre n_j e n_u (T_{ju}), representado em Equação 5.1.

$$T_{iu} = T_{ij} * T_{ju} \tag{5.1}$$

Os valores possíveis de serem assumidos por T_{iu} pertencem ao intervalo $[0, 1]$. No qual o valor 0 indica o máximo de desconfiança e 1 o máximo de confiança entre dois nós. Caso n_i possua um valor de confiança em n_u obtido a partir de uma recomendação anterior, o cálculo de confiança de T_{iu} será a média aritmética entre a confiança anterior (T_{iu}) e o produto da confiança de n_i com n_j (T_{ij}) pela confiança entre n_j e n_u (T_{ju}), conforme mostra a Equação 5.2.

$$T_{iu} = \frac{T_{ij} * T_{ju} + T_{iu}}{2} \tag{5.2}$$

A Tabela 5.1 mostra em quais casos deve ocorrer o cálculo de confiança e quando cada equação será utilizada. Quando n_i não possui confiança em quem recomenda n_j, nenhum cálculo deve ocorrer. Caso contrário, se n_i não possui confiança em n_u, que é o contato recomendado por n_j, a Equação 5.1 será utilizada para determinar T_{iu}. Caso n_i possua confiança em n_u e ela teve origem de uma recomendação, deve-se utilizar a Equação 5.2. Se essa confiança veio de forma direta, ou seja, n_i teve comunicação direta com n_u, nada deve ser feito.

Tabela 5.1: Usos das equações de confiança indireta

Possui confiança em quem recomenda (T_{ij})	**Possui confiança no contato recomendado (T_{iu})**	**Tipo de confiança do contato recomendado**	**Calculo**
Falso	Não interfere	Não interfere	Nada acontece
Verdadeiro	Falso	Não interfere	Equação 5.1
Verdadeiro	Verdadeiro	Recomendação	Equação 5.2
Verdadeiro	Verdadeiro	Direta	Nada acontece

Para essa versão do projeto não foi implementada uma fórmula para calcular a Confiança Direta, caso ela ocorra o nó em questão determina um valor fixo para T_{iu}, que vai variar de acordo com os testes que serão apresentados na Seção 6.2. Suponha o caso em que n_j envia uma mensagem para n_i, a Tabela 5.2 apresenta os casos possíveis para a determinação dessa confiança direta. Caso n_i possua confiança em n_j oriunda de uma recomendação anterior, o valor dela será sobrescrito pelo novo determinado e o tipo será alterado de recomendação para direto. Se o tipo for direto, será apenas alterado o valor da confiança para o novo. Por fim, caso n_i nunca tenha tido contato direto ou indireto com n_j, ele será criado e adicionado na lista de contatos de n_i com sua respectiva confiança determinada.

Tabela 5.2: Operações na confiança direta

Possui confiança em quem enviou a mensagem (T_{ij})	**Tipo de confiança em quem enviou a mensagem**	**Operação**
Verdadeiro	Recomendação	Altera o valor de confiança daquele contato e o tipo de confiança para Direta
Verdadeiro	Direta	Altera o valor de confiança daquele contato
Falso	Não interfere	Adiciona o contato na lista com a confiança determinada

5.2 IMPLEMENTAÇÃO

O sistema deste trabalho segue como base a arquitetura apresentada na Figura 5.1. O primeiro passo é gerar o log das mensagens via o simulador ONE, conforme será apresentado no Capítulo 6. A partir dele é realizada a leitura do arquivo, filtragem e a organização dos dados. Em seguida é calculada a confiança entre cada um dos nós e gerado o conteúdo da transação. Todas as etapas anteriores foram desenvolvidas na linguagem *Python*, versão 3.9.7.

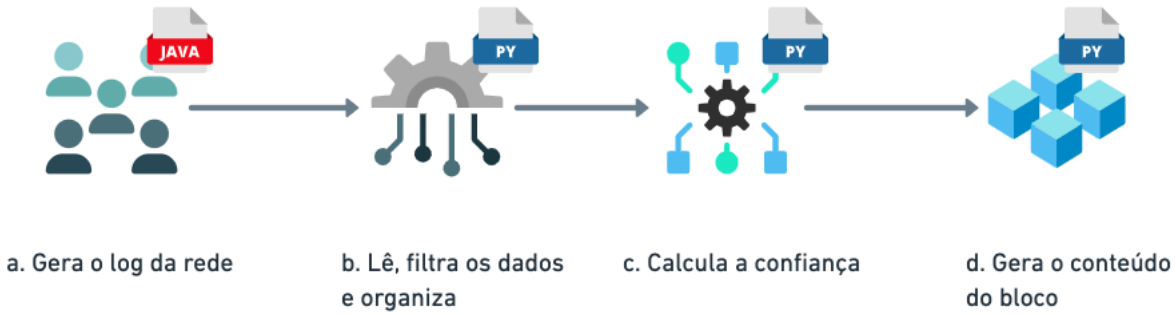


Figura 5.1: Arquitetura do sistema

Foram criadas três classes, para gerenciar localmente os dados recebidos pelo log, que são descritos na Tabela 5.3.

Tabela 5.3: Classes do código

Classe	**Atributos**	**Tipo**	**Descrição**
Trust	*senderID*	*string*	É o ID do nó que enviou a mensagem
	trust	*float*	É o valor de confiança que o destinatário tem em relação à origem
	typeTrust	*string*	É um identificador que determina se a confiança foi de origem direta (*direct*) ou indireta (*recommendation*)
Message	*senderID*	*string*	É o ID do nó que enviou a mensagem
	messageID	*string*	É o ID da mensagem
	deliveryTime	*float*	É o horário de envio da mensagem
TransactionBlock	*receiverID*	*string*	É o ID do nó que recebeu a mensagem
	messages	*array*	É um vetor de mensagens em que cada posição é uma instância do objeto Message
	contacts	*array*	É um vetor de contatos em que cada posição é uma instância do objeto Trust

O Algoritmo 1 é responsável por ler, tratar e agrupar os dados do arquivo de log retornado pelo ONE e o adiciona na variável *logAllMessages* (*l*.2). Os dados estão dentro de um dicionário em que cada chave é o *receiverID* e os valores são uma lista mensagens, conforme o Listing 5.1.

Listing 5.1: Formato e exemplo do dicionário

```
1 Formato:
2 {
3     receiverID: [[messageID, deliveryTime, senderID, typeTrust]]
4 }
5
6 Exemplo:
```

```
{
   c46: [[M1, 1169.2, c42, direct]],
   c49: [[M17, 3220.1, w80, recommendation],
      [M31, 3500.8, c65, direct]]
}
```

Para cada *receiverID* do dicionário é criada uma instância do objeto de transação do bloco *transactionBlockObject* (l.4). É criada uma instância de *messageObject* (l.6) para cada uma das mensagens de *listMessages* que aquele *host* recebeu. Se o tipo dessa mensagem for *direct*, chama o procedimento que calcula a confiança direta (l.8). Senão, chama o procedimento que calcula a confiança indireta (l.11). Essa mensagem é adicionada na lista de *messages* do *transactionBlockObject* (l.12), que ao consumir todas as mensagens é adicionado em *allBlocksList* (l.13).

Algoritmo 1 Gera os blocos

```
1:  procedure GenerateBlocks(allBlocksList)
2:      logAllMessages ← HandleData()
3:      for receiverID, listMessages in logAllMessages do
4:          transactionBlockObject ← TransactionBlock(receiverID)
5:          for message in listMessages do
6:              messageObject ← Message(message.senderID, message.messageID, message.deliveryTime)
7:              if message.type = direct then
8:                  calculateDirectTrust(message, allBlocksList, transactionBlockObject)
9:              else
10:                 if message.type = indirect then
11:                     calculateIndirectTrust(message, allBlocksList, transactionBlockObject)
12:             transactionBlockObject.addMessage(messageObject)
13:         allBlocksList.insert(transactionBlockObject)
```

O Algoritmo 2 lê o arquivo que contém o log e armazena no *dataframe df* (l.2), em seguida ele gera aleatoriamente o tipo de cada mensagem, se será direta ou de recomendação, e coloca em *dfWithTypeTrust* (l.3). A quantidade de mensagens do tipo direta é de 60% e do tipo de recomendação 40% do total. Esses dados são agrupados em *dictGroupedByToHost* (l.4) seguindo como base o Listing 5.1.

Algoritmo 2 Lida com os dados

```
1:  function HandleData( )
2:      df ← ReadCsv("DeliveredMessageReports.txt")
3:      dfWithTypeTrust ← generateTypeTrust(df)
4:      dictGroupedByToHost ← generateListContactsOfEachHostInDict(dfWithTypeTrust)
5:      return dictGroupedByToHost
```

O Algoritmo 3 apresenta o cálculo da confiança direta, que para esse trabalho ela possui um valor fixo (l.2), que varia de acordo com as métricas apresentadas na Seção 6.2. É verificado

se o nó já recebeu alguma mensagem do *senderID* ($l.6$), se sim ele sobrescreve o valor da confiança ($l.9$) e também altera o tipo de confiança caso a anterior tenha sido de *recommendation* ($l.7$). Caso o nó não tenha recebido mensagem do *senderID*, é gerada uma instância *newContact* ($l.11$) da classe *Trust* e ela é adicionada na lista de *contacts* do *transactionBlockObject* ($l.12$).

Algoritmo 3 Calcula a confiança direta

```
1:  procedure CalculateDirectTrust(message, allBlocksList, transactionBlockObject, trust)
2:      senderID ← message.senderID
3:      hasReceivedSenderMessage ← transactionBlockObject.hasReceivedMessage(senderID)
4:      typeTrust ← transactionBlockObject.getTypeTrust(senderID)
5:      if hasReceivedSenderMessage = TRUE then
6:          if typeTrust = recommendation then
7:              transactionBlockObject.setTrustType('direct')
8:          transactionBlockObject.setTrust(trust)
9:      else
10:         newContact ← Trust(message.senderID, trust, 'direct')
11:         transactionBlockObject.addContact(newContact)
```

O Algoritmo 4 é responsável por calcular a confiança indireta. É realizada a busca da *listContactsSender* ($l.3$) do *senderID* ($l.2$) em *allBlocksList*. Em seguida, verificado se o *receiverID* tem alguma confiança no emissor (oriunda de comunicação direta ou de recomendação) e se esse ela é maior que 0 ($l.6$). Caso ele nunca tenha tido contato, a recomendação é ignorada. Do contrário, para cada *contact* dessa lista é verificado se ele já enviou alguma mensagem ao *receiverID* ($l.11$). Caso não seja verdade, a recomendação é calculada ($l.12$) e o contato recomendado é adicionado na lista de contatos ($l.14$). De outro modo, na hipótese de esse contato ter sido proveniente de uma relação direta, a recomendação é ignorada. Contrariamente, é calculada a confiança ($l.17$) e realizada a média com relação ao valor anterior ($l.19$).

Algoritmo 4 Calcula a confiança indireta

```
1:  procedure CalculateIndirectTrust(message, allBlocksList, transactionBlockObject)
2:      senderID ← message.senderID
3:      listContactsSender ← searchSenderTable(senderID, allBlocksList)
4:      hasReceivedSenderMessage ← transactionBlockObject.hasReceivedMessage(senderID)
5:      receiverTrustOnSender ← transactionBlockObject.getTrust(senderID)
6:      if hasReceivedSenderMessage = TRUE and receiverTrustOnSender > 0 then
7:          for contact in listContactsSender do
8:              hasReceivedContactMessage ← transactionBlockObject.hasReceivedMessage(contact.senderID)
9:              typeTrustContact ← transactionBlockObject.getTypeTrust(contact.senderID)
10:             receiverTrustOnContact ← transactionBlockObject.getTrust(contact.senderID)
11:             if hasReceivedContactMessage = FALSE then
12:                 trust ← contact.trust * receiverTrustOnSender
13:                 newContact ← Trust(contact.senderID, trust, 'recommendation')
14:                 transactionBlockObject.addContact(newContact)
15:             else
16:                 if typeTrustContact = recommendation then
17:                     trust ← contact.trust * receiverTrustOnSender
18:                     trust ← (trust + receiverTrustOnContact)/2
19:                     transactionBlockObject.setTrust(trust)
```

5.3 RESUMO

Este capítulo apresentou o sistema proposto para este trabalho. Dentro disso, mostrou as bases teóricas para o cálculo de confiança indireta. Por fim, a implementação foi exibida e explicada com os principais algoritmos do sistema.

6 AVALIAÇÃO

Este trabalho implementou o sistema proposto no Capítulo 5. Afim de realizar a simulação de uma rede DTN, foi utilizado o simulador *The Opportunistic Network Environment simulator (ONE)*. O simulador foi especificamente projetado para avaliar roteamento DTN e protocolos de aplicação. Ele permite que os usuários criem cenários baseados em diferentes modelos de movimento sintético e traços do mundo real e oferece uma estrutura para a implementação de protocolos de roteamento e aplicativos (já incluindo seis protocolos de roteamento bem conhecidos). As ferramentas de visualização interativa e pós-processamento suportam a avaliação de experimentos e um modo de emulação permite que o ONE se torne parte de um ambiente de teste DTN do mundo real, já que seu mapa é do centro da cidade de Helsinki na Finlândia (Keränen et al., 2009). Um exemplo de como é a interface com o aplicativo está representado na Figura 6.1.

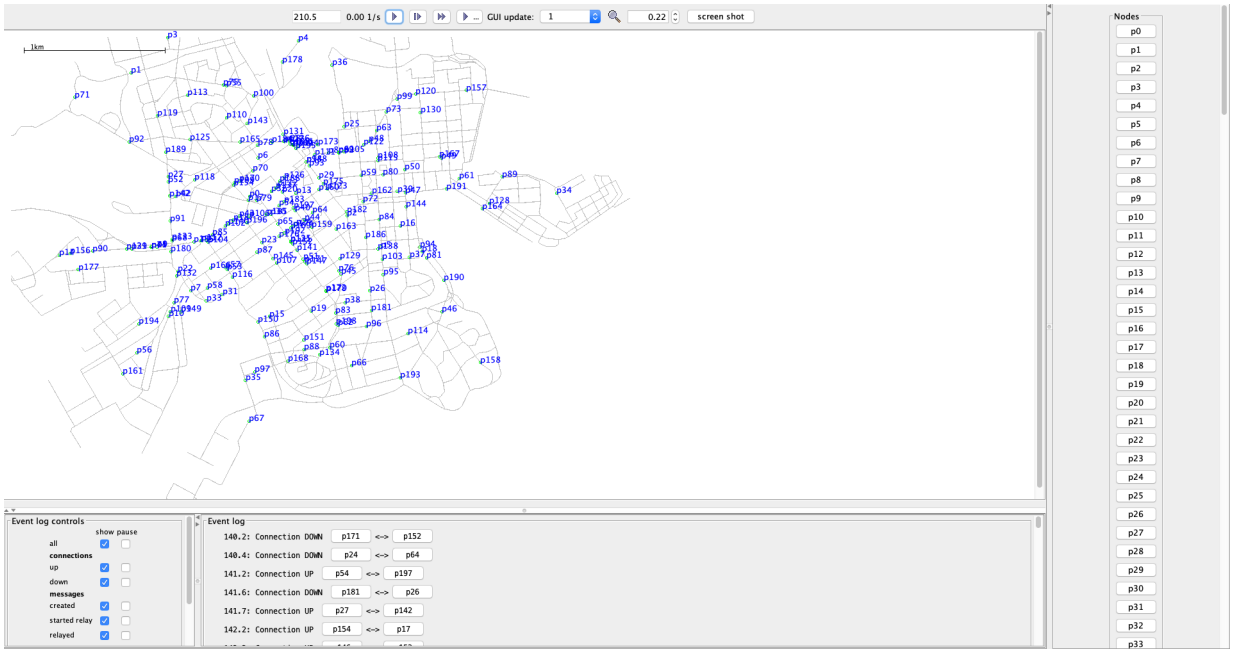


Figura 6.1: Exemplo do aplicativo ONE

O cenário de movimentação do simulador foi configurado com os parâmetros apresentados na Tabela 6.1. Para essa simulação foi considerado o tamanho padrão do mapa de 4500x3400 metros. Os nós podem se mover apenas pelas ruas e o movimento utilizado é o mesmo para todos eles, onde um nó escolhe um ponto aleatório do mapa, calcula o caminho mínimo para ele com o algoritmo de *Dijkstra* e se move em velocidade constante que varia de 0.5 até 1.5 m/s, que seria a velocidade de uma pessoa andando. É utilizado o modelo de roteamento *Epidemic*, no qual depois que dois nós trocam as mensagens entregáveis, ele tenta trocar também todas as outras mensagens até que ambos os nós tenham o mesmo conjunto de mensagens ou a conexão

seja interrompida. Com a finalidade de entender como o sistema se comporta com variadas quantidades de nós, foram gerados dados para 10, 20, 30, 40, 50, 100, 150 e 200 nós. Para cada um deles foram considerados os tempos de simulação de 5000 e 60000 segundos.

Tabela 6.1: Parâmetros do ONE

Parâmetro	**Valor**
Tempo de simulação	5000 e 60000 s
Tamanho do mapa	4500x3400 m
Número de grupos de nós	1
Número de nós	10, 20, 30, 40, 50, 100, 150 e 200
Protocolo de roteamento	Epidemic
Velocidade mínima do nó	0.5 m/s
Velocidade máxima do nó	1.5 m/s
Modelo de movimento	ShortestPathMapBasedMovement

No contexto deste trabalho ele foi utilizado unicamente para gerar os dados de mensagens entregues *(DeliveredMessagesReport)*, pois o objetivo aqui é pegar históricos de mensagens entre os nós. Para todas as trocas de mensagens é possível acessar diversas propriedades, as quais são relevantes para essa aplicação apenas: tempo (*time*), id da mensagem (ID), id da origem (*fromHost*) e id do destino (*toHost*). Um exemplo do formato desses dados está representado na Tabela 6.2.

Tabela 6.2: Exemplo de dados

ID	**time**	**fromHost**	**toHost**
M29	131.8000	p94	p97
M1	1169.2000	p42	p94
M23	938.9000	p94	p97

6.1 AMBIENTE DE DESENVOLVIMENTO

O ambiente de desenvolvimento utilizado na avaliação do sistema proposto consistiu no emprego da ferramenta ONE, versão 1.6.0, conforme comentado na Seção 5.2. Além disso, a ferramenta foi utilizada no sistema operacional macOS Catalina, versão 10.15.7. O desenvolvimento ocorreu nos ambientes de desenvolvimento integrado (do inglês, *Integrated Development Environment* - IDE) do *Visual Studio Code*, versão 1.63.1, e Eclipse, versão 4.20.0. Em que o primeiro foi utilizado para programar com a linguagem *Python* e o segundo para rodar o simulador ONE. Durante as simulações, dados estatísticos foram coletados e armazenados em arquivos de texto.

6.2 METODOLOGIA

A metodologia de avaliação deste trabalho consistiu em gerar diversos cenários no simulador ONE, dos quais foram alterados apenas os parâmetros: tempo simulação, número de nós e valor da confiança direta, nos quais seus respectivos valores seguiram como referência a Tabela 6.3.

Tabela 6.3: Valores de parâmetros analisados

Nós	**Tempo (s)**	**Valor da confiança direta**
10, 20, 30, 40, 50, 100, 150 e 200	5000 e 60000	0.5, 0.7, 0.9 e aleatório

Ao total foram gerados 64 casos a serem analisados. Para cada um, foram gerados os tipos de resultados apresentados na Tabela 6.4. A primeira métrica visa apresentar a quantidade de mensagens trocadas ao longo do tempo antes da execução, a partir dos dados de entrada. A segunda mostra a relação entre a quantidade de mensagens aceitas ao longo do tempo depois da execução do programa. A terceira métrica tem a finalidade de explicar os motivos pelos quais a quantidade de mensagens aceitas é diferente das esperadas. Para analisar a distribuição dos valores de confiança no decorrer do tempo, foi criada a quarta métrica. A confiança entre dois indivíduos pode ser alterada ao longo do tempo e o gráfico anterior apresenta apenas o último valor de confiança. Por isso, foi criada a métrica cinco que visa mostrar quantas vezes o valor de cada tipo de confiança mudou no decorrer da execução do programa, válido para todos os *hosts*. Com a finalidade de ter uma representação visual das conexões entre os nós e seus respectivos valores de confiança, foi criado um grafo. Complementar ao item seis, o sétimo é uma tabela que visa comparar a conexão de cada *host* com todos os outros em relação à média daquele atributo, que pode ser direta, indireta ou sem conexão.

Tabela 6.4: Métricas de avaliação

	Tipo	**Objetivo**	**Dados**
1.	Gráfico de linhas	Analisar a quantidade de mensagens com relação ao tempo esperadas de acordo com os dados de entrada. As mensagens podem ser de tipo direta ou de recomendação.	Eixos: Quantidade de mensagens x Tempo Linhas: Direta e Recomendação Fonte: Log das mensagens de entrada
2.	Gráfico de linhas	Analisar a quantidade de mensagens com relação ao tempo obtidas com sucesso de acordo com os dados de saída do programa. As mensagens podem ser de tipo direta ou de recomendação.	Eixos: Quantidade de mensagens x Tempo Linhas: Direta e Recomendação Fonte: Lista de blocos
3.	Gráfico de barras	Apresentar os motivos pelos quais o que era esperado no gráfico 1 ter divergido do que realmente aconteceu no gráfico 2.	Barras: Nenhum contato e sem confiança
4.	Gráfico de dispersão	Analisar os valores de confiança (direta e indireta) obtidos no decorrer do tempo.	Eixos: Confiança x Tempo
5.	Gráfico de barras	Apresentar a quantidade de vezes que o valor de confiança foi alterado para cada *host*, que é o motivo pelo qual o gráfico 4 não começa em 0.	Barras: Direta e recomendação
6.	Grafo	Apresentar o grafo de confiança gerado após a execução do programa.	Vértices: Nome do *host* Arestas: Valor da confiança Fonte: Lista de blocos
7.	Tabela	Apresentar para cada *host* a quantidade de conexões diretas, indiretas e sem conexão entre todos os membros da rede. Assim como a média de cada um desses valores.	Colunas: Nome do *host*, quantidade de conexões diretas, indiretas e sem conexão Linhas: Dados dos *hosts* e a média dos dados

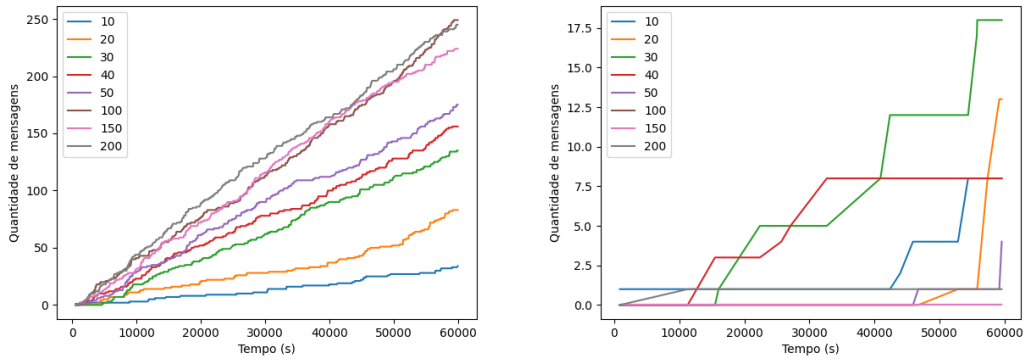
6.3 RESULTADOS

Esta seção apresenta os resultados obtidos nos casos explorados com base nas métricas definidas anteriormente. Analisando apenas as mudanças de número de nós e tempo de simulação, chegamos em 16 cenários. A Subseção 6.3.1 apresenta, para 8 desses cenários, a relação entre o tempo e a quantidade de mensagens diretas e indiretas após a execução do programa. Como são muitos cenários, aqui vamos discutir apenas três deles. Dos quais, a Subseção 6.3.2 corresponde a 10 nós no tempo de 60000s. Em contrapartida a Subseção 6.3.3 possui 200 nós e tempo de 5000s. Por fim, a Subseção 6.3.4 tem 200 nós no tempo de 60000s.

6.3.1 Variação na quantidade de nós

Essa subseção consiste em analisar para o tempo fixo de 60000 segundos, a quantidade de mensagens diretas e indiretas aceitas para todas as variações de nós. Para o caso das mensagens diretas representado na Figura 6.2 (a), podemos observar a consistência em quando aumenta o número de nós a quantidade de mensagens também cresce. As únicas exceções são 100 nós, que possui a maior quantidade de mensagens e 150 que tem menos mensagens do que 100 e 200.

Em contrapartida, na Figura 6.2 (b) que dentre todas as variações na quantidade de nós, apenas 6 cenários possuíram alguma conexão indireta. Além disso, conclui-se que o caso que mais possui recomendações foi 30 nós, seguido por 20, 40, 10, 50 e 200. Assim, percebe-se a inconsistência com relação ao aumento do número de nós e a quantidade de mensagens. Adicionalmente, conforme dito anteriormente, a quantidade de mensagens de recomendações enviadas correspondem a 40% do total. Por exemplo, se analisarmos minuciosamente o caso de 30 nós, a quantidade de mensagens diretas são de aproximadamente 150 e as indiretas 18. Podemos perceber a quantidade de mensagens de recomendação aceitas correspondem apenas a 7% do total. Isso significa que há uma grande taxa de rejeição de recomendação. Além de ficar mais evidente nos outros cenários, que possuem ainda mais nós.



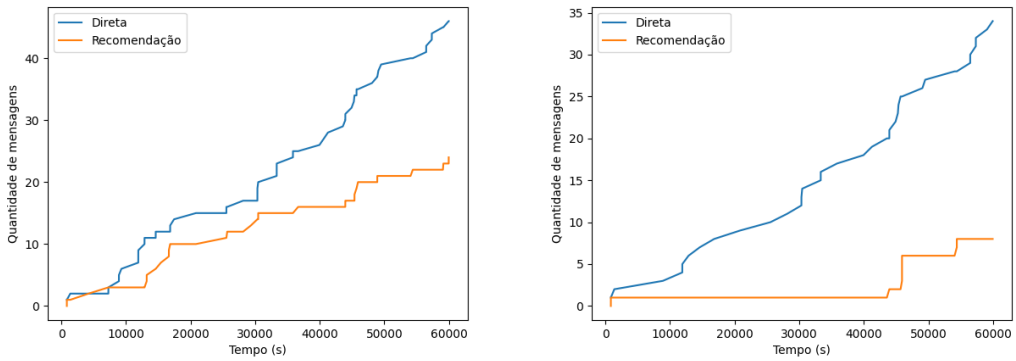
(a) Direta

(b) Recomendação

Figura 6.2: Quantidade de mensagens trocadas após a execução

6.3.2 Cenário 1

Esse cenário consiste em 10 nós se comunicando durante o tempo de 60000 segundos. Podemos perceber pela Figura 6.3 (a) que a quantidade de mensagens diretas esperada antes da execução era de mais de 40 e de recomendação 20, aproximadamente. Após a execução do programa, na Figura 6.3 (b), podemos estimar uma diminuição de 10 mensagens em contatos diretos e uma diferença de 15 mensagens a menos no de recomendação. Além disso, podemos observar que o número de recomendações aceitas aumentam próximo ao tempo de 45000s, o que indica que é necessário ter uma troca de mensagens diretas antes de aceitar as indiretas.



(a) Antes da execução

(b) Depois da execução

Figura 6.3: Quantidade de mensagens trocadas

Essa queda da quantidade de mensagens antes e depois da execução no caso da confiança indireta está representada na Figura 6.4. Em que podemos perceber 13 casos onde não foi possível aceitar a comunicação pois o nó que recebeu a mensagem nunca tinha tido contato com quem enviou a recomendação. Outro ponto de ressalva é que esse cenário não caiu no caso em que o valor de confiança entre o destinatário e remetente é zero.

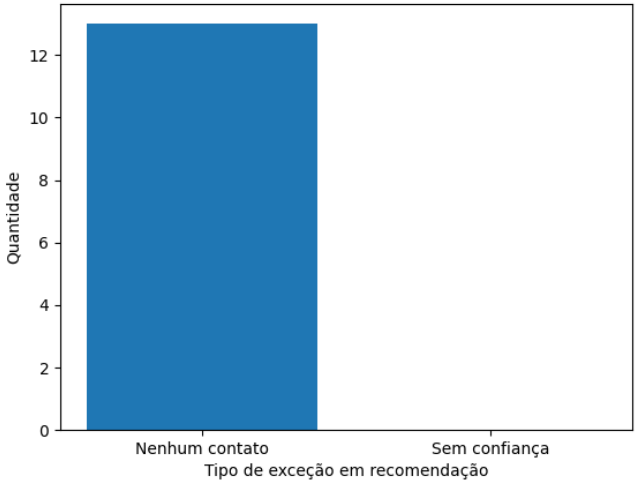


Figura 6.4: Casos de mensagens não aceitas

A Figura 6.5 representa o motivo da divergência na confiança direta entre Figura 6.3 (a) e Figura 6.3 (b). Os casos em que um nó recebeu novamente um contato direto de um outro conhecido, o valor da confiança é alterado. Como o processo de montagem do gráfico da Figura 6.3 (b) considera apenas o último contato entre os nós, todos os contatos anteriores a ele não são considerados. Além disso, é possível perceber que houve nós que se comunicaram mais de uma vez.

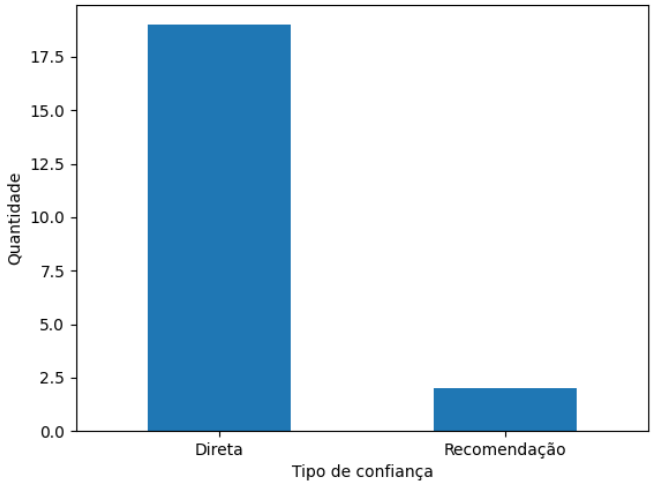


Figura 6.5: Quantidade de vezes que alterou o valor da confiança

A Figura 6.6 apresenta o grafo das conexões de todos os nós da rede ao final da execução. Em que as linhas rosas e azuis são representações de uma conexão direta e indireta, respectivamente. Nesse caso podemos perceber que foi representado um cenário em que o valor da confiança direta é de 0.5. Além disso, é possível perceber que os nós que possuem contato de recomendação são p4-p6, p7-p6, p8-p6, p0-p7, p0-p8, p7-p8, p5-p7 e p1-p5. É possível concluir que o p6 foi um nó muito recomendado pelos outros.

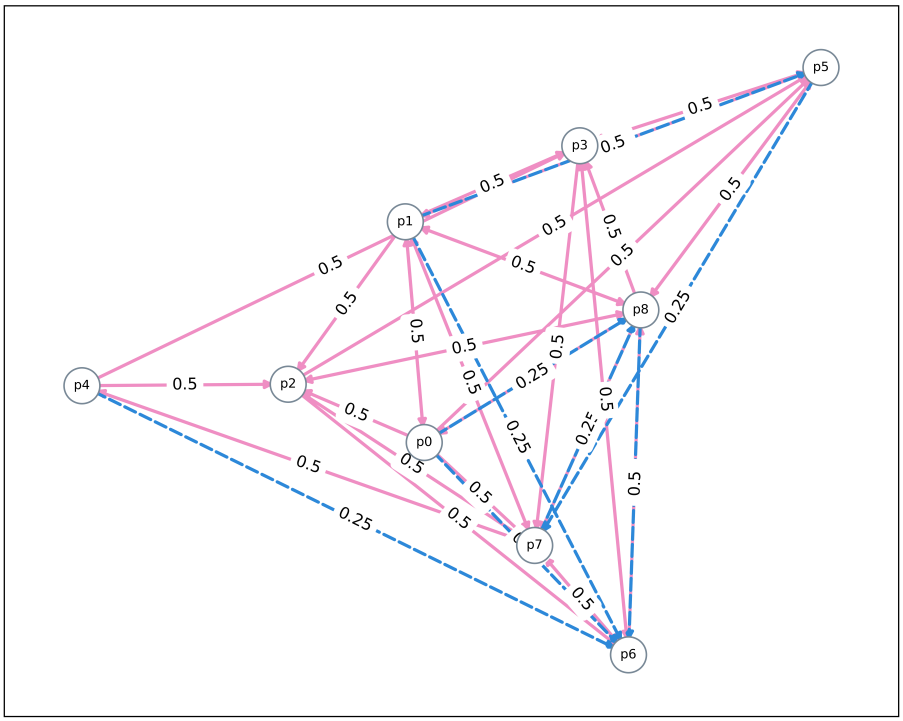


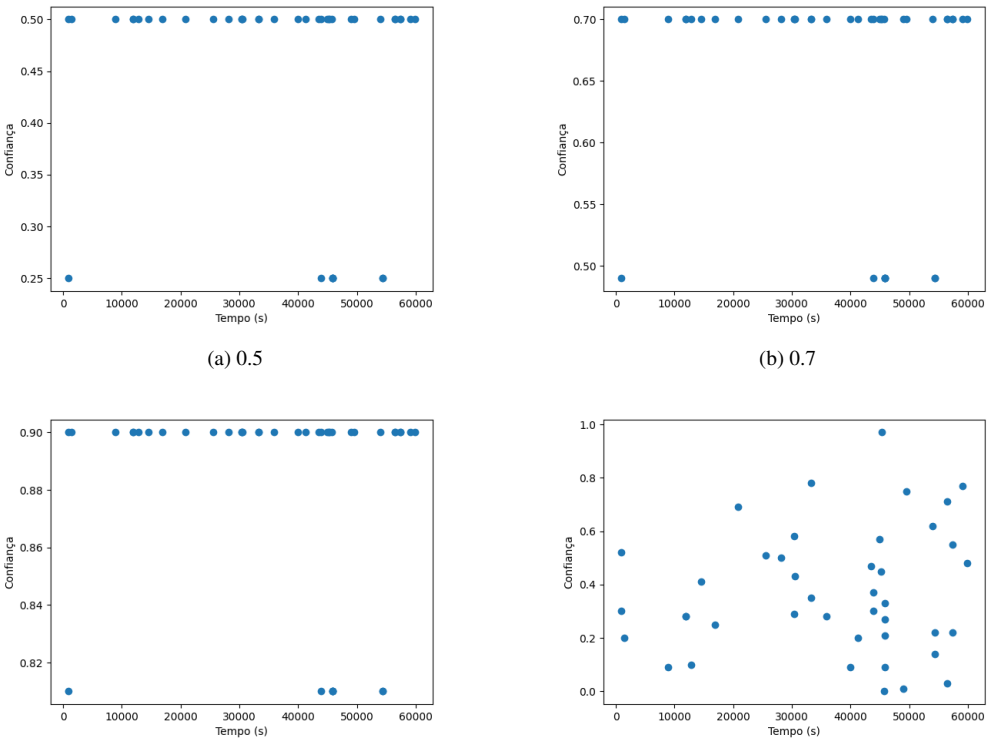
Figura 6.6: Grafo das conexões entre os nós

A Figura 6.7 é uma representação tabular do grafo anterior. Vê-se a consistência em relação as conexões do nó p6 e a quantidade de conexões de recomendação. Além disso, podemos perceber a desproporcionalidade das conexões em relação a média. Por exemplo, a média de recomendação é de 0.86, e apenas 4 nós a realizaram, sendo que o nó p6 foi responsável por 50% desse valor. Com isso também é possível de analisar no caso da direta, em que o nó p7 realizou a maior quantidade.

Host	Direta	Recomendação	Sem contato
p0	2.0	0.0	6.0
p1	4.0	0.0	4.0
p2	4.0	0.0	4.0
p3	5.0	0.0	3.0
p4	1.0	0.0	7.0
p5	4.0	1.0	3.0
p6	3.0	4.0	1.0
p7	7.0	1.0	0.0
p8	4.0	2.0	2.0
Média	3.78	0.89	3.33

Figura 6.7: Conexões entre os nós

Os gráficos da Figura 6.8 são representações da distribuição dos valores de confiança para as quatro tipos de alterações de confiança direta. Para os casos em que o valor da confiança é fixo, podemos perceber que as Figuras (a), (b) e (c) possuem o mesmo comportamento e que os valores começam a dispersar um pouco mais a partir do instante 45000s, que são os valores da confiança indireta. Já para os valores aleatórios na Figura 6.8 (d), observa-se o espalhamento dos valores e não é possível analisar a diferença entre os pontos diretos e indiretos.



(a) 0.5

(b) 0.7

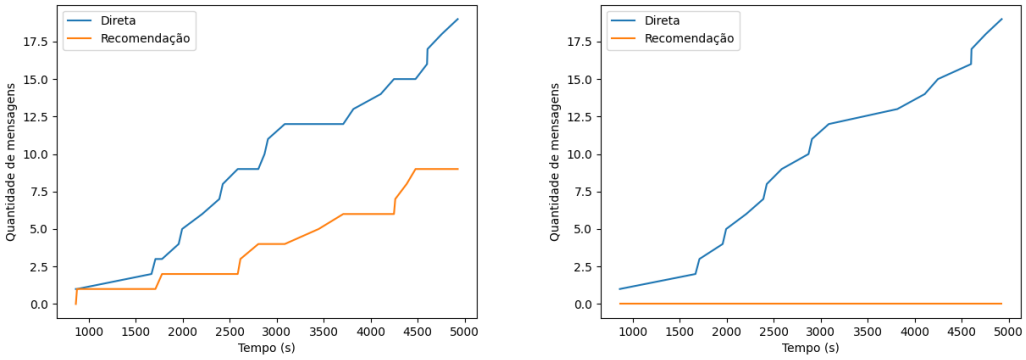
(c) 0.9

(d) Aleatório

Figura 6.8: Distribuição dos valores de confiança ao longo do tempo

6.3.3 Cenário 2

Esse cenário consiste em 200 nós se comunicando durante o tempo de 5000 segundos. Podemos perceber pela Figura 6.9 (a) que a quantidade de mensagens diretas esperada antes da execução era de mais de 17 e de recomendação 8, aproximadamente. Após a execução do programa, na Figura 6.9 (b), podemos perceber que a direta não sofreu alteração e que não houve caso de recomendação bem sucedida.



(a) Antes da execução

(b) Depois da execução

Figura 6.9: Quantidade de mensagens trocadas

Podemos observar pela Figura 6.10 que mais de 8 recomendações foram negadas pois o nó não obteve contato anterior com quem compartilhou a tabela de confiança.

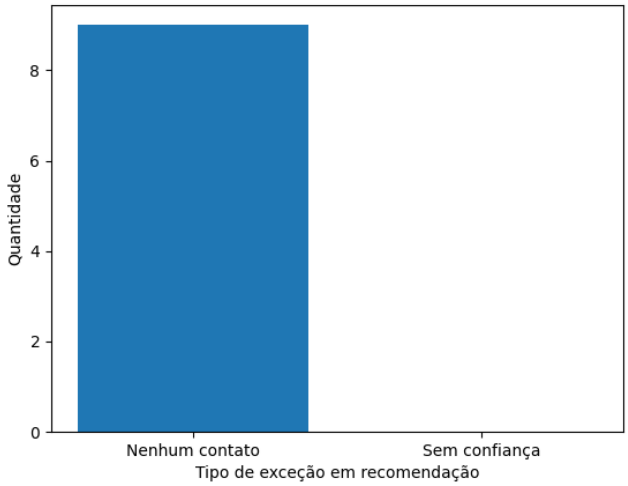


Figura 6.10: Casos de mensagens não aceitas

A Figura 6.11 mostra o motivo pelo qual todas as mensagens esperadas de confiança direta foram constantes em relação às obtidas. Pois não teve nenhuma alteração no valor de confiança direta e nem indireta. Isso também mostra que nenhum nó teve contato mais de um contato com outro.

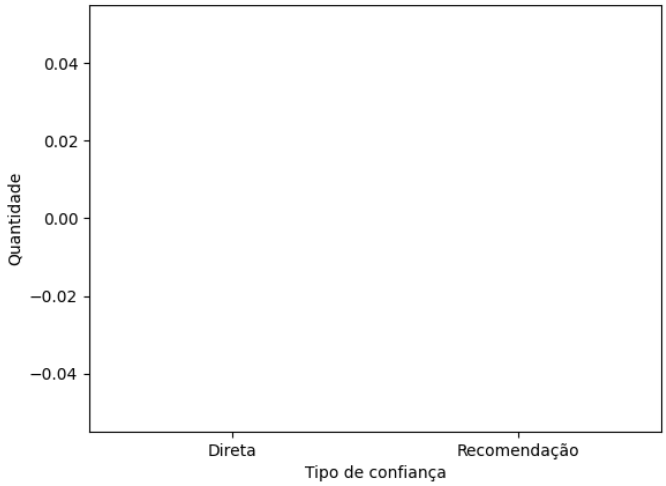


Figura 6.11: Quantidade de vezes que alterou o valor da confiança

Nota-se que a rede da Figura 6.12 ficou bem dispersa, na qual os nós tiveram apenas conexões com um nó. Isso mostrou uma enorme diferença com o cenário anterior, em que o tempo foi muito maior e a quantidade de *hosts* muito menor. Uma possível conclusão para isso seria que, apesar da quantidade de nós disponíveis, o tempo foi insuficiente para que os contatos pudessem ter sido realizados com sucesso.

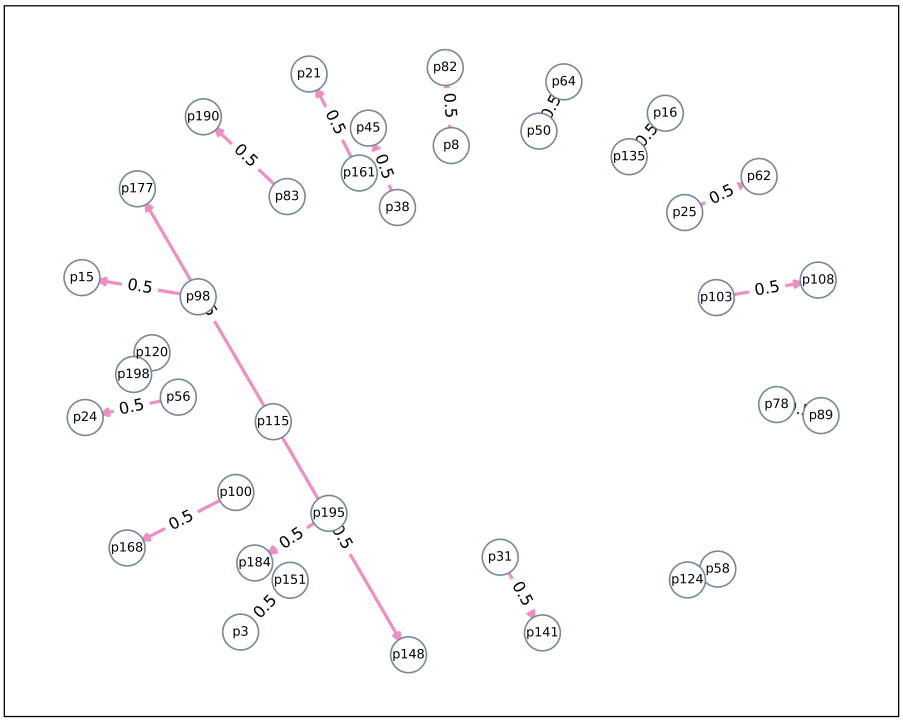


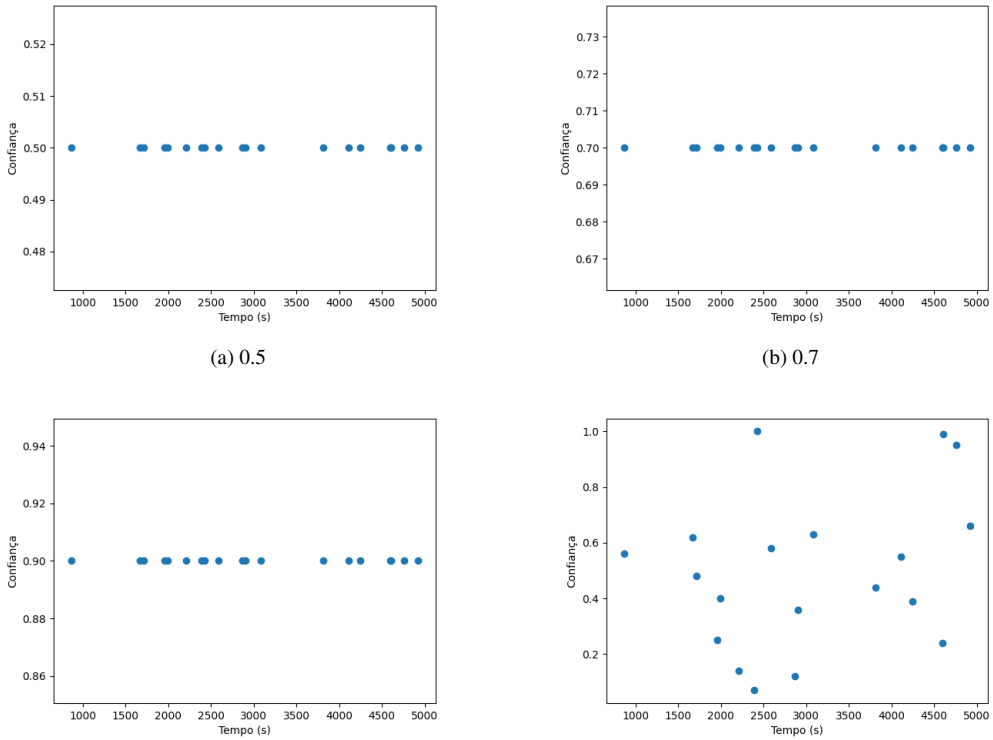
Figura 6.12: Grafo das conexões entre os nós

A Figura 6.13 mostra que apesar de ter 200 nós disponíveis, apenas 27 se comunicaram na simulação. Outro ponto interessante é a média de 0.7 mensagens diretas. Apenas 8 ficaram abaixo dela e não realizaram contato direto e o resto fez um contato.

Host	Direta	Recomendação	Sem contato
p108	1.0	0.0	25.0
p109	0.0	0.0	26.0
p124	1.0	0.0	25.0
p14	0.0	0.0	26.0
p141	1.0	0.0	25.0
p144	0.0	0.0	26.0
p148	1.0	0.0	25.0
p15	1.0	0.0	25.0
p151	0.0	0.0	26.0
p16	1.0	0.0	25.0
p164	0.0	0.0	26.0
p168	1.0	0.0	25.0
p177	1.0	0.0	25.0
p184	1.0	0.0	25.0
p190	1.0	0.0	25.0
p198	1.0	0.0	25.0
p21	1.0	0.0	25.0
p24	1.0	0.0	25.0
p3	1.0	0.0	25.0
p33	0.0	0.0	26.0
p45	1.0	0.0	25.0
p57	0.0	0.0	26.0
p62	1.0	0.0	25.0
p64	1.0	0.0	25.0
p82	1.0	0.0	25.0
p84	0.0	0.0	26.0
p89	1.0	0.0	25.0
Média	0.7	0.0	25.3

Figura 6.13: Conexões entre os nós

Na Figura 6.14 observa-se a constância no valor da confiança nos casos em que o valor direto é fixo e varia em 0.5, 0.7 e 0.9. Em contrapartida houve um grande espalhamento na Figura 6.14 (d). Além disso, os valores iniciam em 1000s, assim como estimado na Figura 6.9, isso se dá pelo motivo que não houve caso de alteração no valor da confiança (Figura 6.11), isto é, mais de uma comunicação entre os nós.



(a) 0.5

(b) 0.7

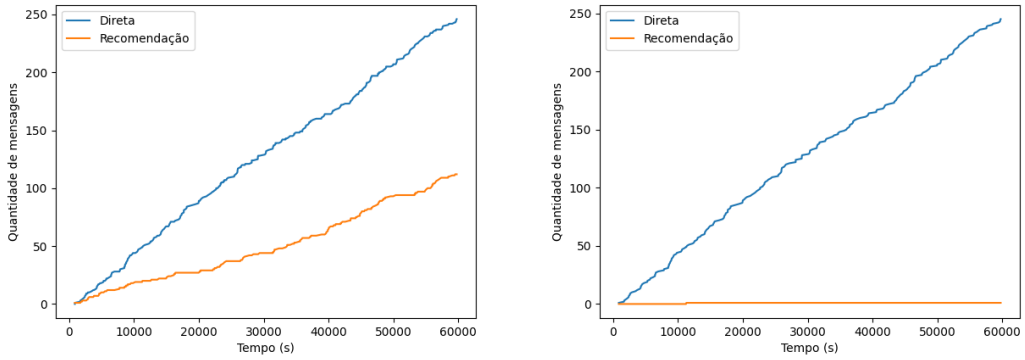
(c) 0.9

(d) Aleatório

Figura 6.14: Distribuição dos valores de confiança ao longo do tempo

6.3.4 Cenário 3

Esse cenário consiste em 200 nós se comunicando durante o tempo de 60000 segundos. Podemos perceber pela Figura 6.15 (a) que a quantidade de mensagens diretas esperada antes da execução era de mais de 250 e de recomendação 100, aproximadamente. Após a execução do programa, na Figura 6.15 (b), podemos perceber que a direta não sofreu alteração e que houve um pouco mais do que 0 recomendações bem sucedidas.



(a) Antes da execução

(b) Depois da execução

Figura 6.15: Quantidade de mensagens trocadas

Apesar de ter sido um cenário com muitos nós no maior tempo, pudemos perceber uma grande quantidade de rejeição de recomendação. O motivo dessa recusa foi mais de 100 indicações terem sido feitas por um nó que não tinha contato anterior com o recomendador (Figura 6.16).

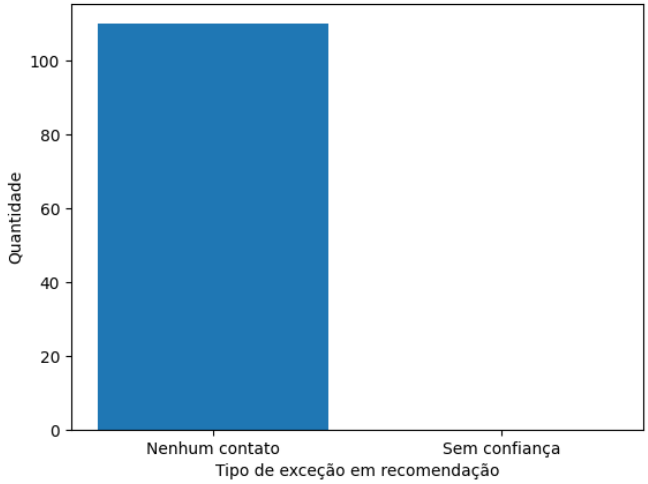


Figura 6.16: Casos de mensagens não aceitas

Na Figura 6.17 nota-se houve apenas um caso em que o valor da confiança direta alterou. Isso significa que de 200 nós em 60000 segundos, aconteceu apenas uma situação em que um nó se comunicou duas vezes com outro.

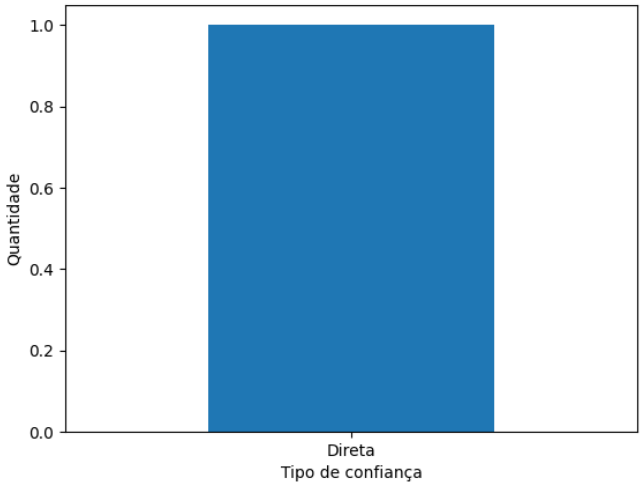


Figura 6.17: Quantidade de vezes que alterou o valor da confiança

O grafo de conexão entre os nós está legível, por conta da sobreposição dos dados. Por conta disso, será possível analisar apenas a tabela da Figura 6.18. Consta-se que apenas o nó p127 possui uma conexão indireta. Apesar da simulação ter 200 nós, apenas 169 trocaram mensagens com sucesso.

Host	Direta	Recomendação	Sem contato
p0	1.0	0.0	167.0
p1	2.0	0.0	166.0
p10	1.0	0.0	167.0
p100	4.0	0.0	164.0
p101	3.0	0.0	165.0
p102	2.0	0.0	166.0
p103	2.0	0.0	166.0
p104	1.0	0.0	167.0
p105	1.0	0.0	167.0
p106	1.0	0.0	167.0
p107	0.0	0.0	168.0
p108	3.0	0.0	165.0
p109	1.0	0.0	167.0
p11	0.0	0.0	168.0
p110	3.0	0.0	165.0
p112	1.0	0.0	167.0
p113	1.0	0.0	167.0
p114	3.0	0.0	165.0
p115	3.0	0.0	165.0
p116	1.0	0.0	167.0
p118	1.0	0.0	167.0
p119	1.0	0.0	167.0
p12	1.0	0.0	167.0
p120	3.0	0.0	165.0
p121	1.0	0.0	167.0
p122	0.0	0.0	168.0
p123	0.0	0.0	168.0
p124	1.0	0.0	167.0
p125	1.0	0.0	167.0
p126	1.0	0.0	167.0
p127	3.0	1.0	164.0
p128	1.0	0.0	167.0
p129	0.0	0.0	168.0
p13	2.0	0.0	166.0
p130	0.0	0.0	168.0
p132	0.0	0.0	168.0
p135	0.0	0.0	168.0
p136	3.0	0.0	165.0
p137	2.0	0.0	166.0
p138	1.0	0.0	167.0
p139	1.0	0.0	167.0
p14	1.0	0.0	167.0
p141	0.0	0.0	168.0
p142	1.0	0.0	167.0
p143	1.0	0.0	167.0
p144	0.0	0.0	168.0
p145	0.0	0.0	168.0
p146	2.0	0.0	166.0
p147	1.0	0.0	167.0
p148	2.0	0.0	166.0
p15	1.0	0.0	167.0
p150	1.0	0.0	167.0
p151	3.0	0.0	165.0
p152	2.0	0.0	166.0
p154	1.0	0.0	167.0
p155	1.0	0.0	167.0
p156	0.0	0.0	168.0
p157	3.0	0.0	165.0
p158	2.0	0.0	166.0
p159	1.0	0.0	167.0
p16	2.0	0.0	166.0
p162	1.0	0.0	167.0
p163	3.0	0.0	165.0
p164	2.0	0.0	166.0
p165	1.0	0.0	167.0
p166	2.0	0.0	166.0
p167	0.0	0.0	168.0
p168	2.0	0.0	166.0
p169	1.0	0.0	167.0
p17	2.0	0.0	166.0
p170	1.0	0.0	167.0
p171	4.0	0.0	164.0
p172	1.0	0.0	167.0
p173	4.0	0.0	164.0
p174	2.0	0.0	166.0
p175	1.0	0.0	167.0
p176	2.0	0.0	166.0
p177	2.0	0.0	166.0
p178	2.0	0.0	166.0
p180	1.0	0.0	167.0
p181	2.0	0.0	166.0
p184	1.0	0.0	167.0
p185	1.0	0.0	167.0
p186	3.0	0.0	165.0
p187	2.0	0.0	166.0

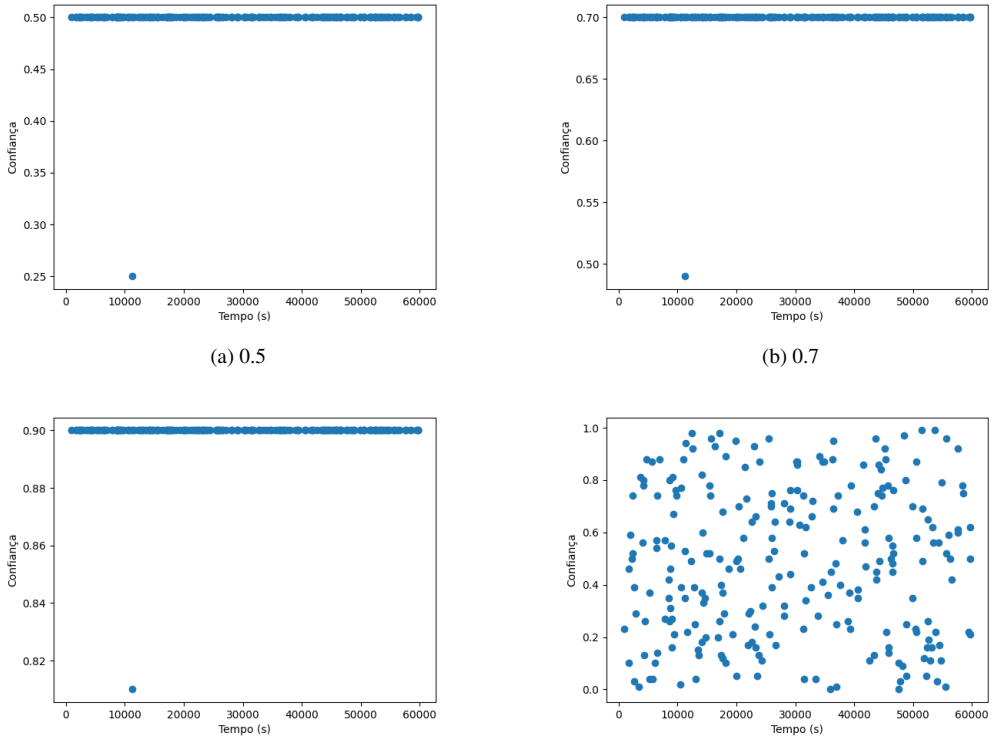
(a) Primeira metade

p189	3.0	0.0	165.0
p19	2.0	0.0	166.0
p190	1.0	0.0	167.0
p192	3.0	0.0	165.0
p193	4.0	0.0	164.0
p194	2.0	0.0	166.0
p196	1.0	0.0	167.0
p198	2.0	0.0	166.0
p2	1.0	0.0	167.0
p21	3.0	0.0	165.0
p22	1.0	0.0	167.0
p23	1.0	0.0	167.0
p24	3.0	0.0	165.0
p25	0.0	0.0	168.0
p27	1.0	0.0	167.0
p28	1.0	0.0	167.0
p29	0.0	0.0	168.0
p3	2.0	0.0	166.0
p31	1.0	0.0	167.0
p32	0.0	0.0	168.0
p33	1.0	0.0	167.0
p34	1.0	0.0	167.0
p35	4.0	0.0	164.0
p36	0.0	0.0	168.0
p38	1.0	0.0	167.0
p39	3.0	0.0	165.0
p4	3.0	0.0	165.0
p40	1.0	0.0	167.0
p41	4.0	0.0	164.0
p42	1.0	0.0	167.0
p43	1.0	0.0	167.0
p44	1.0	0.0	167.0
p45	1.0	0.0	167.0
p46	1.0	0.0	167.0
p47	0.0	0.0	168.0
p48	0.0	0.0	168.0
p49	3.0	0.0	165.0
p5	1.0	0.0	167.0
p50	0.0	0.0	168.0
p51	1.0	0.0	167.0
p52	2.0	0.0	166.0
p53	0.0	0.0	168.0
p54	1.0	0.0	167.0
p56	3.0	0.0	165.0
p57	2.0	0.0	166.0
p58	1.0	0.0	167.0
p59	1.0	0.0	167.0
p6	1.0	0.0	167.0
p60	1.0	0.0	167.0
p61	1.0	0.0	167.0
p62	4.0	0.0	164.0
p64	2.0	0.0	166.0
p65	0.0	0.0	168.0
p66	1.0	0.0	167.0
p68	3.0	0.0	165.0
p69	2.0	0.0	166.0
p7	0.0	0.0	168.0
p70	2.0	0.0	166.0
p71	2.0	0.0	166.0
p73	1.0	0.0	167.0
p74	3.0	0.0	165.0
p75	1.0	0.0	167.0
p76	1.0	0.0	167.0
p77	0.0	0.0	168.0
p79	1.0	0.0	167.0
p80	1.0	0.0	167.0
p81	1.0	0.0	167.0
p82	1.0	0.0	167.0
p83	1.0	0.0	167.0
p84	3.0	0.0	165.0
p85	2.0	0.0	166.0
p86	2.0	0.0	166.0
p87	1.0	0.0	167.0
p89	1.0	0.0	167.0
p9	3.0	0.0	165.0
p90	1.0	0.0	167.0
p91	2.0	0.0	166.0
p92	1.0	0.0	167.0
p93	0.0	0.0	168.0
p94	2.0	0.0	166.0
p95	1.0	0.0	167.0
p97	0.0	0.0	168.0
p98	1.0	0.0	167.0
p99	0.0	0.0	168.0
Média	1.45	0.01	166.54

(b) Segunda metade

Figura 6.18: Conexões entre os nós

Pode-se perceber na Figura 6.19 que há muito mais pontos do que no cenário anterior. Nos casos em que o valor da confiança direta é constante, é visível que há sempre um ponto afastado, que é a confiança vinda da recomendação. E como nos cenários anteriores, o gráfico aleatório está com valores bem distribuídos.



(a) 0.5

(b) 0.7

(c) 0.9

(d) Aleatório

Figura 6.19: Distribuição dos valores de confiança ao longo do tempo

6.4 RESUMO

Este capítulo apresentou uma avaliação do sistema proposto no Capítulo 5. Com a utilização de métricas e uma metodologia definida para os três cenários, observou-se a consistência do sistema, assim como as particularidades de cada cenário. Foi mostrado que o sucesso da confiança indireta se dá pelas trocas de mensagens anteriores de um nó em relação ao outro. Conclui-se que o que mais interfere para que ocorra a conexão indireta é um tempo de simulação relativamente alto, pois isso dá a chance de ter mais comunicabilidade. Mas, ele deve ser crescer proporcionalmente a quantidade de nós daquela rede. Como pudemos observar, para o cenário 1 o tempo foi satisfatório. Entretanto, para as situações 2 e 3 ele teria que ser bem maior, para trazer mais resultados aceitáveis.

7 CONCLUSÃO

Nas redes DTN os modelos de gerenciamento de confiança são importantes para mitigar os problemas que nós maliciosos ou egoístas podem causar no encaminhamento de mensagens. Existem diversos trabalhos publicados que tratam desse problema. Entretanto, nenhum deles propôs um modelo de gerenciamento de confiança para redes tolerantes ao atraso com o uso da tecnologia *Blockchain*.

Este trabalho propôs um sistema de gerenciamento de confiança por recomendação para redes tolerantes ao atraso e desconexão. Além disso, é sugerido um modelo de utilização do *Blockchain* para que cada nó da rede realize o cálculo e o armazenamento das confianças de modo distribuído. A proposta é genérica e pode ser aplicada a qualquer tipo de rede que se encaixe nesse cenário. Foram estudados diversos casos com parâmetros distintos, tais como o tamanho da rede, o tempo de simulação e a interferência das relações diretas. Concluiu-se que a taxa de mensagens de recomendações aceitas pelo sistema sofre influência da frequência com que o nó destinatário comunicou-se com o remetente. Além disso, ele mostrou consistência em todos os casos testados com relação a taxa de aceitação de mensagens diretas e indiretas. De modo que, praticamente todas as mensagens diretas são aceitas e há um filtro das mensagens de recomendação.

7.1 TRABALHOS FUTUROS

Entre as possibilidade de trabalhos futuros destaca-se o emprego de um modelo de confiança direta. Isso irá agregar muito tendo em vista a importância e o uso da comunicação direta. Além disso, seria interessante testar outras diferentes porcentagens de quantidade de mensagens diretas e indiretas. Afim de analisar o impacto de cada uma na rede e como o sistema se comportará. Poderia também evoluir o modelo de cálculo de confiança indireta proposto neste trabalho. Atualmente utiliza-se apenas um nível da recomendação, adicionar n níveis possibilitaria ter acesso a mais indicações de nós.

Um trabalho a ser explorado é a integração do sistema com o *Blockchain* global, para validar como eles se comportam em conjunto com as redes DTN, qual a eficiência e sincronia com relação ao tempo de publicação de blocos anteriores e o conteúdo tabela a ser atualizada. Outro estudo importante seria a adição de nós maliciosos e egoístas, para analisar como a rede se comporta e a efetividade da segurança. Ademais, um trabalho futuro poderia ser implementar o sistema em dispositivos do mundo mundo real.

REFERÊNCIAS

Agarwal, A., Starobinski, D. e Little, T. D. (2011). Phase transition of message propagation speed in delay-tolerant vehicular networks. *IEEE Transactions on Intelligent Transportation Systems*, 13(1):249–263.

Amendola, D., De Rango, F., Massri, K. e Vitaletti, A. (2014). Efficient neighbor discovery in rfid based devices over resource-constrained dtn networks. Em *2014 IEEE International Conference on Communications (ICC)*, páginas 3842–3847. IEEE.

Amin, R., Ripplinger, D., Mehta, D. e Cheng, B.-N. (2015). Design considerations in applying disruption tolerant networking to tactical edge networks. *IEEE Communications Magazine*, 53(10):32–38.

Asuquo, P., Chikezie, A. e Stephen, B. (2021). A decentralised trust framework for emergency communication using dtn. *INTERNATIONAL JOURNAL OF ENGINEERING RESEARCH & TECHNOLOGY (IJERT)*, 10(12):801–823.

Bauwens, M., Kostakis, V. e Pazaitis, A. (2019). *Peer to Peer*. University of Westminster Press.

Bhaskar, N. D. e Chuen, D. L. K. (2015). Bitcoin mining technology. Em *Handbook of digital currency*, páginas 45–65. Elsevier.

Blaze, M., Feigenbaum, J. e Lacy, J. (1996). Decentralized trust management. Em *Proceedings 1996 IEEE Symposium on Security and Privacy*, páginas 164–173.

Bodkhe, U., Tanwar, S., Parekh, K., Khanpara, P., Tyagi, S., Kumar, N. e Alazab, M. (2020). Blockchain for industry 4.0: A comprehensive review. *IEEE Access*, 8:79764–79800.

Cho, H.-H., Chen, C.-Y., Shih, T. K. e Chao, H.-C. (2014). Survey on underwater delay/disruption tolerant wireless sensor network routing. *IET Wireless Sensor Systems*, 4(3):112–121.

Cho, J.-H., Chan, K. e Adali, S. (2015). A survey on trust modeling. *ACM Comput. Surv.*, 48(2):28:1–28:40.

De Andrade, D. e Albini, L. C. P. (2016). Fully distributed public key management through digital signature chains for delay and disrupt tolerant networks. Em *2016 IEEE 13th International Conference on Mobile Ad Hoc and Sensor Systems (MASS)*, páginas 316–324. IEEE.

Ekblaw, A., Azaria, A., Halamka, J. D. e Lippman, A. (2016). A case study for blockchain in healthcare:“medrec” prototype for electronic health records and medical research data. Em *Proceedings of IEEE open & big data conference*, volume 13, página 13.

Fill, H.-G. e Härer, F. (2020). Usage scenarios for blockchain technologies in the domain of civil law notaries. *International Trends of Legal Informatics, Festschrift für Erich Schweighofer, Editions Weblaw, Bern*.

Grandison, T. e Sloman, M. (2000). A survey of trust in internet applications. *IEEE Communications Surveys and Tutorials*, 3:2–16.

Greca, R. e Albini, L. C. P. (2018). Truman: Trust management for vehicular networks. Em *2018 IEEE Symposium on Computers and Communications (ISCC)*, páginas 00801–00806.

Jiang, P., Guo, F., Liang, K., Lai, J. e Wen, Q. (2020). Searchain: Blockchain-based private keyword search in decentralized storage. *Future Generation Computer Systems*, 107:781–792.

Kaisar, S. (2021). Emergency response and post-disaster recovery using smartphone-based applications. Em *Digital Services in Crisis, Disaster, and Emergency Situations*, páginas 31–49. IGI Global.

Karaarslan, E. e Adiguzel, E. (2018). Blockchain based dns and pki solutions. *IEEE Communications Standards Magazine*, 2(3):52–57.

Keränen, A., Ott, J. e Kärkkäinen, T. (2009). The ONE Simulator for DTN Protocol Evaluation. Em *SIMUTools '09: Proceedings of the 2nd International Conference on Simulation Tools and Techniques*, New York, NY, USA. ICST.

Kouicem, D. E., Imine, Y., Bouabdallah, A. e Lakhlef, H. (2020). A decentralized blockchain-based trust management protocol for the internet of things. *IEEE Transactions on Dependable and Secure Computing*.

Lee, H. e Yeon, C. (2021). Blockchain-based traceability for anti-counterfeit in cross-border e-commerce transactions. *Sustainability*, 13(19):11057.

Li, Q. e Cao, G. (2011). Mitigating routing misbehavior in disruption tolerant networks. *IEEE transactions on information forensics and security*, 7(2):664–675.

Maesa, D. D. F. e Mori, P. (2020). Blockchain 3.0 applications survey. *Journal of Parallel and Distributed Computing*, 138:99–114.

Maesa, D. D. F., Mori, P. e Ricci, L. (2017). Blockchain based access control. Em *IFIP international conference on distributed applications and interoperable systems*, páginas 206–220. Springer.

McCorry, P., Shahandashti, S. F. e Hao, F. (2017). A smart contract for boardroom voting with maximum voter privacy. Em *International Conference on Financial Cryptography and Data Security*, páginas 357–375. Springer.

Misaghi, M., da Silva, E. e Albini, L. C. P. (2012). Distributed self-organized trust management for mobile ad hoc networks. Em *International Conference on Networked Digital Technologies*, páginas 506–518. Springer.

Mu, B. e Yuan, S. (2010). A method for evaluating initial trust value of direct trust and recommender trust. Em *2010 International Conference On Computer Design and Applications*, volume 2, páginas V2–185–V2–190.

Nagrath, P., Aneja, S. e Purohit, G. (2019). Attacks in delay tolerant networks: Classification and analysis. Em *2019 11th International Conference on Communication Systems & Networks (COMSNETS)*, páginas 489–491. IEEE.

Naik, N. e Jenkins, P. (2020). uport open-source identity management system: An assessment of self-sovereign identity and user-centric data platform built on blockchain. Em *2020 IEEE International Symposium on Systems Engineering (ISSE)*, páginas 1–7. IEEE.

Nakamoto, S. (2008). Bitcoin: A peer-to-peer electronic cash system. *Decentralized Business Review*, página 21260.

Nofer, M., Gomber, P., Hinz, O. e Schiereck, D. (2017). Blockchain. *Business & Information Systems Engineering*, 59(3):183–187.

Okupski, K. (2014). Bitcoin developer reference. Em *Eindhoven*.

Oliveira, C., Moreira, M., Rubinstein, M., Costa, L. e Duarte, O. (2007). Redes tolerantes a atrasos e desconexões. *Minicursos do XXV SBRC*, página 203–256.

O'Dair, M. e Beaven, Z. (2017). The networked record industry: How blockchain technology could transform the record industry. *Strategic Change*, 26(5):471–480.

Park, Y., Sur, C. e Rhee, K.-H. (2018). A secure incentive scheme for vehicular delay tolerant networks using cryptocurrency. *Security and Communication Networks*, 2018.

Pires, T. P. (2016). Tecnologia blockchain e suas aplicações para provimento de transparência em transações eletrônicas. `https://bdm.unb.br/bitstream/10483/16252/1/2016_TimoteoPimentaPires_tcc.pdf`. Acessado em 01/12/2021.

Rodrigues, J. J. e Soares, V. N. (2015). An introduction to delay and disruption-tolerant networks (dtns). Em *Advances in Delay-Tolerant Networks (DTNs)*, páginas 1–21. Elsevier.

Roy, S., Basu, S. e Chowdhury, S. (2021). Blockchain leveraged node incentivization in cooperation-based delay tolerant networks. Em *Opportunistic Networks*, páginas 187–206. CRC Press.

Tanenbaum, A., Wetherall, D. e Translations, O. (2011). *Redes de computadores*. PRENTICE HALL BRASIL.

Tornell, S. M., Calafate, C. T., Cano, J.-C. e Manzoni, P. (2015). Dtn protocols for vehicular networks: An application oriented overview. *IEEE Communications Surveys Tutorials*, 17(2):868–887.

Tovar, A., Friesen, T., Ferens, K. e McLeod, B. (2010). A dtn wireless sensor network for wildlife habitat monitoring. Em *CCECE 2010*, páginas 1–5. IEEE.

Truong, N., Jayasinghe, U., Um, T.-W. e Lee, G. M. (2016). A survey on trust computation in the internet of things. *THE JOURNAL OF KOREAN INSTITUTE OF COMMUNICATIONS AND INFORMATION SCIENCES (J-KICS)*, 33:10–27.

Wu, X. e Liang, J. (2021). A blockchain-based trust management method for internet of things. *Pervasive and Mobile Computing*, 72:101330.

Xu, J., Xue, K., Tian, H., Hong, J., Wei, D. S. e Hong, P. (2020). An identity management and authentication scheme based on redactable blockchain for mobile networks. *IEEE Transactions on Vehicular Technology*, 69(6):6688–6698.

Yazdinejad, A., Parizi, R. M., Dehghantanha, A., Karimipour, H., Srivastava, G. e Aledhari, M. (2020). Enabling drones in the internet of things with decentralized blockchain-based security. *IEEE Internet of Things Journal*, 8(8):6406–6415.

Zhou, L., Wang, L. e Sun, Y. (2018). Mistore: a blockchain-based medical insurance storage system. *Journal of medical systems*, 42(8):1–17.